

Formal Methods and Tool-suites for CPS Security, Safety and Verification

Lectures 3 and 4: Motion Planning and Controls with Safety and Temporal Constraints

John S. Baras

Institute for Systems Research and Dept. of Electr. and Comp. Engin.


University of Maryland College Park, USA,

and ACCESS Centre Royal Instit. of Technology (KTH), Stockholm, Sweden,

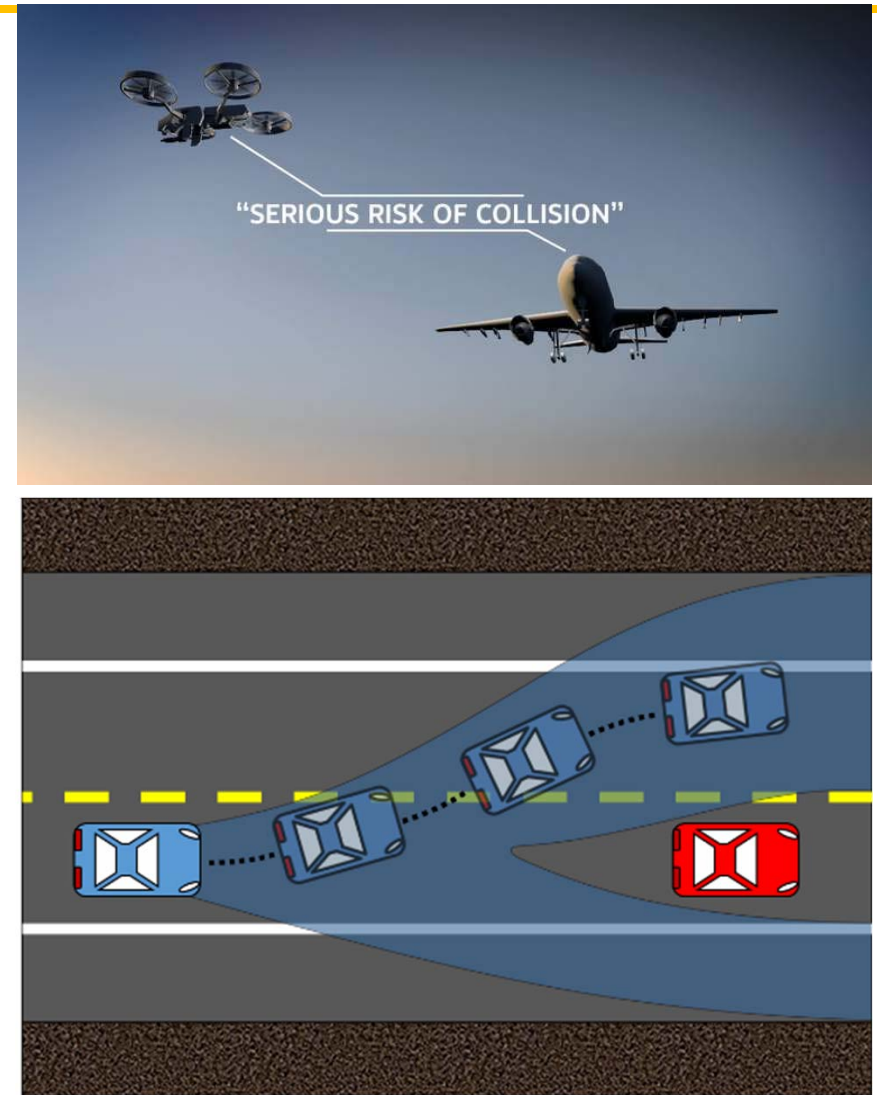
and Instit. for Advanced Study Technical Univ. of Munich (TUM), Germany

August 2, 3, 2018

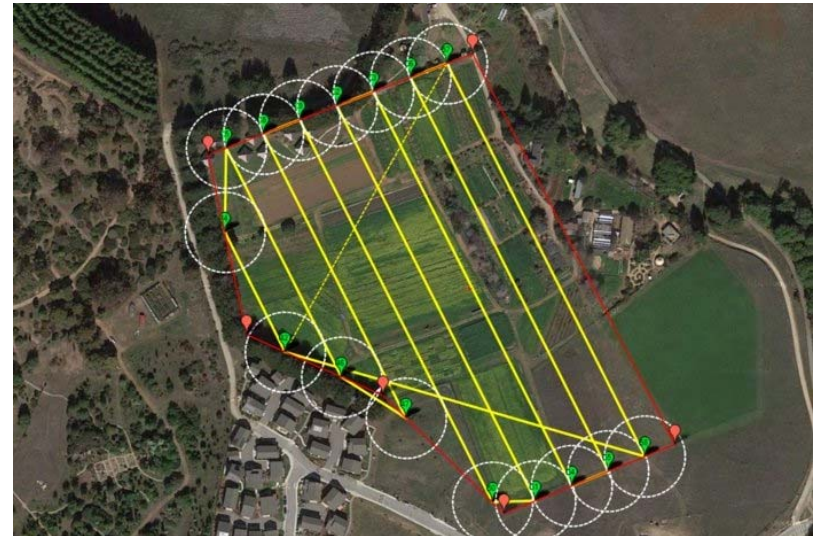
**MARKTOBERDORF INTERNATIONAL SUMMER SCHOOL ON ENGINEERING SECURE AND
DEPENDABLE SOFTWARE SYSTEMS**

- Motion (or trajectory) planning
 - High level task specifications  low-level control commands executable on the plant
- Key problem in CPS
- Plant: autonomous vehicle, UAV, (humanoid) robot, industrial robotic arm
- Human-machine interactions
- Safety and temporal constraints

- Safety
 - UAVs in commercial airspace
 - Autonomous vehicles & human-driven cars
- Human involvement
 - Safety is critical and fundamental
- Physical limitation
 - To avoid states that lead to unavoidable collision



- Synthesize plan from task specifications
 - Agriculture monitor
 - Security and surveillance
 - Search and rescue
 - Disaster relief / Emergency communications
- Perform task in an optimal manner with given time constraints



- **Safety** in Motion Planning
 - Methods to **prove** the **safety** of the plan and the trajectory tracking controls
 - **Control synthesis** algorithms to generate safe control bounds in collision avoidance scenarios
- **Temporal Requirements** in Motion Planning
 - Two methods to solve the motion planning problem with **timed temporal constraints**
 - Convert the temporal constraints to an optimization problem
 - Convert the constraints to timed automata

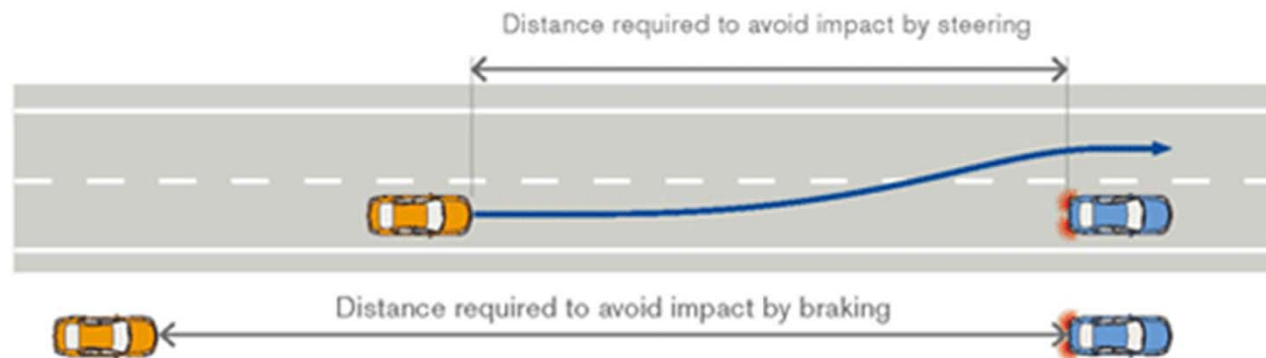
- **Part I:** Reachable set based safety verification and control synthesis
 - I.1 Reachable set based verification
 - I.2 Control synthesis using optimization
- **Part II:** Motion planning for temporal logics with finite time constraints
 - II.1 Mixed integer optimization based method
 - II.2 Timed automata based method
- **Part III:** Event –Triggered Controller Synthesis for Dynamical Systems with Temporal Logic Constraints
- **Part V:** Event –Triggered Feedback Control for Signal Temporal Logic Tasks

Part I: Reachable Set Based Safety Verification and Control Synthesis

Part I.1: Reachable Set Based Verification

Reachable Set Based Verification

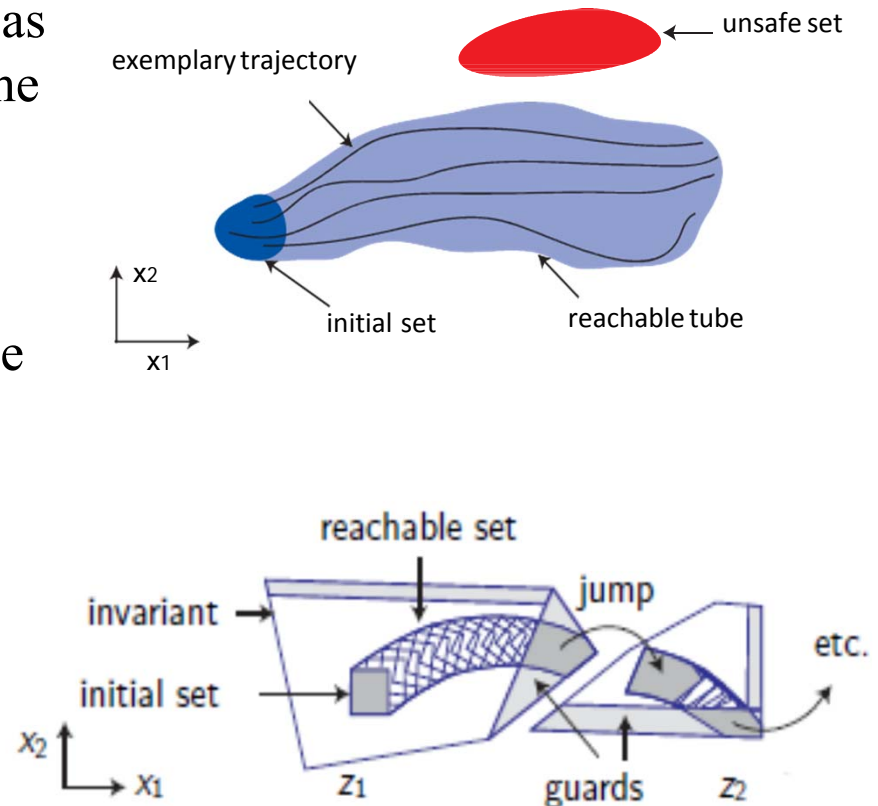
- **Verification** of the **safety** of the motion planner and the trajectory tracking controls for UAV¹ and autonomous car.
 - Reference **trajectories**
 - Trajectory tracking **controls**
 - Q: How to prove **safety** of the system given **sensor noise**, **control disturbance** and **dynamics** of the system?



1. J. Moschler, Y. Zhou J. S. Baras and J. Joh, "A System Engineering Approach to Collaborative Coordination of UAS in the NAS with Safety Guarantees", Proceedings of 2013 Integrated Communications Navigation and Surveillance (ICNS) Conference, Herndon, Virginia, pp. U1-1~U1-12, April 8~10, 2014.

Reachability Set and Safety

- Reachable set of a dynamics is defined as set of states reachable at a particular time from a **bounded initial set**, a **control set** and a **disturbance set**.
- The **occupancy** set of a vehicle can be computed from the reachable set and the size of vehicle.
- Efficient reachable set computation normally uses convex approximations such as **ellipsoids** [A. B. Kurzhansk 2000] and **zonotopes** [M. Althoff et al 2008, A. Girard et al 2006].
- The control sets can be **synthesized collaboratively** so that the reachable sets of the UAVs have no intersection. [Zhou and Baras CDC 2015]



Reachable Sets: Linear and Affine Systems

- Lateral dynamics of autonomous vehicle

$$\begin{aligned}\dot{x}(t) &= A(t)x(t) + B(t)u(t) + v(t), \\ x(0) &\in \mathcal{X}_0, u(t) \in \mathcal{U}(t), v(t) \in \mathcal{V}\end{aligned}$$

- Trajectory tracking controller based on MPC

$$u(t) = -K(t)x(t) + h(\gamma(t), t)$$

- Overall linear system

$$\dot{x}(t) = A_c(t)x(t) + \underbrace{v(t) + B(t)h(\gamma(t), t)}_{w(t) \in \mathcal{W}(t)}$$

Recursive Computation of Reachable Set

$$\mathcal{R}(\delta, \mathcal{X}_0) = \Phi_c(\delta, 0)\mathcal{X}_0 \oplus \mathcal{R}(\delta, \{0\}).$$

$$V = \mathcal{R}(\delta, \{0\})$$

$A_c(t) = A(t) - B(t)K(t)$ is the matrix of the closed loop system.

$\Phi_c(t, s)$ is the transition matrix for the closed-loop system.

reachability tube the union of Ω_i , $\Omega_{i+1} \leftarrow \Phi_i \Omega_i \oplus V$.

$\Phi_i = \Phi_c((i+1)\delta, i\delta)$, $V = \mathcal{R}(\delta, \{0\})$ δ is the sampling rate

Computation of Reachable Set

Algorithm 1 Reachable Set for Linear System

Input: Initial set \mathcal{X}_0 , horizon T , sampling step N .

Output: Reachable set $\mathcal{R}([0, T], \mathcal{X}_0)$

- 1: $\delta \leftarrow T/N$
 - 2: $r_\delta \leftarrow \sup_{w(t) \in \mathcal{W}(t)} \int_0^\delta \Phi_c(t, \tau) w(\tau) d\tau$
 - 3: $V \leftarrow \mathcal{B}(r_\delta)$ $\triangleright \mathcal{R}(\delta, \{0\})$ overapproximated by V
 - 4: $s_\delta \leftarrow \sup_{x \in \mathcal{X}_0} (\Phi_c(\delta, 0) - I - A_c(\delta))x$
 - 5: $\Omega_0 \leftarrow CH(\mathcal{X}_0 \cup \Phi_0 \mathcal{X}_0) \oplus \mathcal{B}(s_\delta + r_\delta)$ $\triangleright \mathcal{R}([0, \delta], \mathcal{X}_0)$ overapproximated by Ω_0
 - 6: **for** i from 0 to $N - 2$ **do**
 - 7: $\Omega_{i+1} \leftarrow \Phi \Omega_i \oplus V$ \triangleright Computationally expensive
 - 8: **end for**
 - 9: **return** $\bigcup_{i \in \{0, \dots, N-1\}} \Omega_i$
-

$$\Omega_i = \left(\prod_{j=0}^i \Phi_j \right) \Omega_0 \oplus \left[\bigoplus_{j=0}^{i-1} \left(\prod_{k=0}^j \Phi_k \right) V \right]$$

Iterative Over-approximation

$$\left. \begin{array}{l} \mathcal{A}_i \leftarrow \Phi_i \mathcal{A}_{i-1} \\ \mathcal{S}_i \leftarrow \mathcal{S}_{i-1} \oplus V_{i-1} \\ V_i \leftarrow \Phi_{i-1} V_{i-1} \\ \Omega_i \leftarrow \mathcal{A}_i \oplus \mathcal{S}_i \end{array} \right\} i = 1, 2, \dots, N$$

where the induction is initialized by, $\mathcal{A}_0 \leftarrow \Omega_0, V_0 \leftarrow V, \mathcal{S}_0 \leftarrow \{0\}$.

Extensions to Hybrid Automata with Linear Dynamics

(Hybrid Automata). A hybrid automaton H is a tuple $H = (Q, X, Flow, \mathcal{I}, Trans, S_0)$ defined by the following components:

- Q : a finite set of discrete locations q_i , which are vertices of a graph (V, E) whose edges are discrete transition defined by $Trans$.
- X : a finite number of continuous state variables $[x_1, \dots, x_{\|X\|}] \in R^{\|X\|}$ whose dynamics is defined by $Flow$ of every location q_i .
- $Flow$: a set of predicates on the derivatives of continuous variable, more specifically, $Flow(q)$ specifies the set for allowed $X \times \dot{X}$. The particular flow condition we are used here are general nonlinear systems in the form of $\dot{x} = f(x, w)$, for $w \in \mathcal{W}$. If the underlying dynamics at each location is linear then convex \mathcal{W} is assumed.

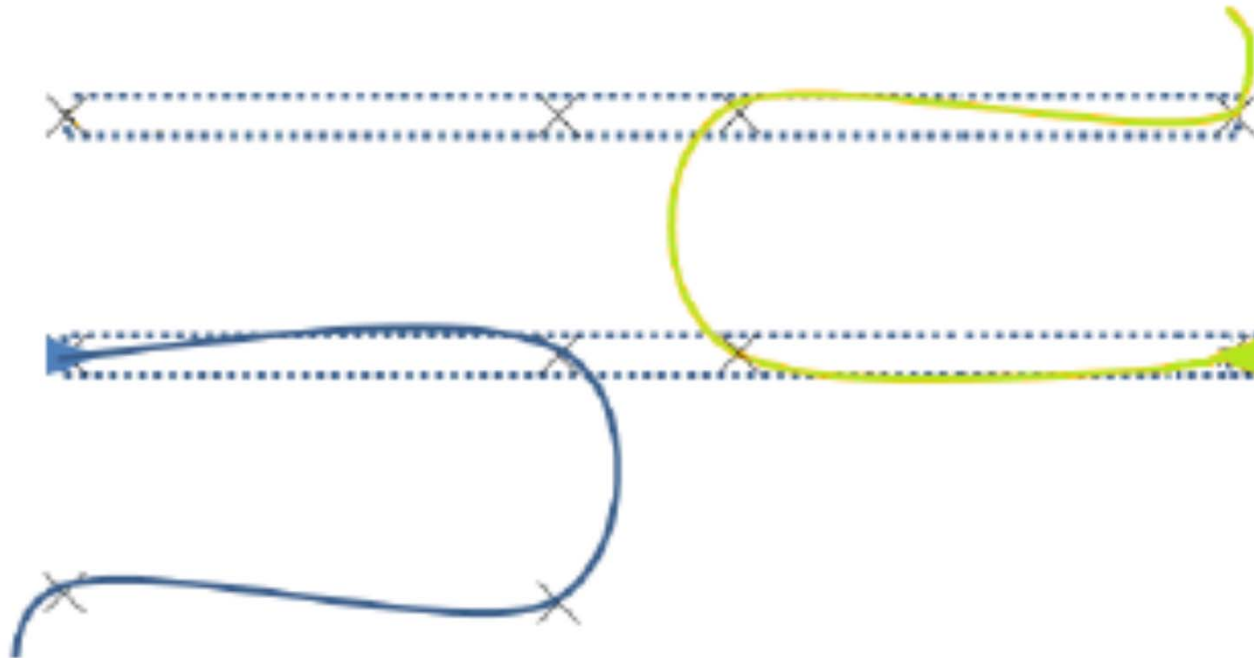
Extensions to Hybrid Automata with Linear Dynamics

- \mathcal{I} : invariant set is a set of predicates on the continuous variables X for each location. If the underlying dynamics is linear then convex invariant constraint is assumed.
- Trans : is defined by a set of transition $E \subseteq Q \times Q$ and its associated jump predicate G on X and jump function μ . The transition from q to q' is taken, if there exists a transition from q to q' where its predicate G is satisfied, then next state $x' = \mu(x)$. Denote the set of state satisfying G as \mathcal{G} .
- S_0 : a set of initial state $Q_0 \times \mathcal{X}_0 \subset Q \times X$.

Computation using reachable set based methods has been demonstrated for real-time applications of autonomous vehicles with high degrees of nonlinearity

The outcome of the reachability analysis includes reachable sets of position and orientation of autonomous aircraft along with piloted aircraft that are inside the surveying workspace. When there are nearby friendly aircraft which are autonomously controlled by the same high level control system, the reachability set data is shared and safety is verified by checking that the reachable sets are not colliding in any instant of the time. The UAV operator will be able to obtain the reachable tubes of the autonomous aircraft in almost real-time and transmit location and trajectory data to Air Traffic Control (ATC) and nearby aircraft if desired.

Results and Examples



Planned trajectory. The blue and green arrows mark the initial positions of the UAVs. The crosses marks the waypoints for individual UAVs in a survey tasks. Possible information is marked by doted area. The corresponding curve are reference trajectory generated by the planner.

Verification for Collision Avoidance Controller

- Lateral dynamics of autonomous vehicle

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) + v(t),$$

$$x(0) \in \mathcal{X}_0, u(t) \in \mathcal{U}(t), v(t) \in \mathcal{V}$$

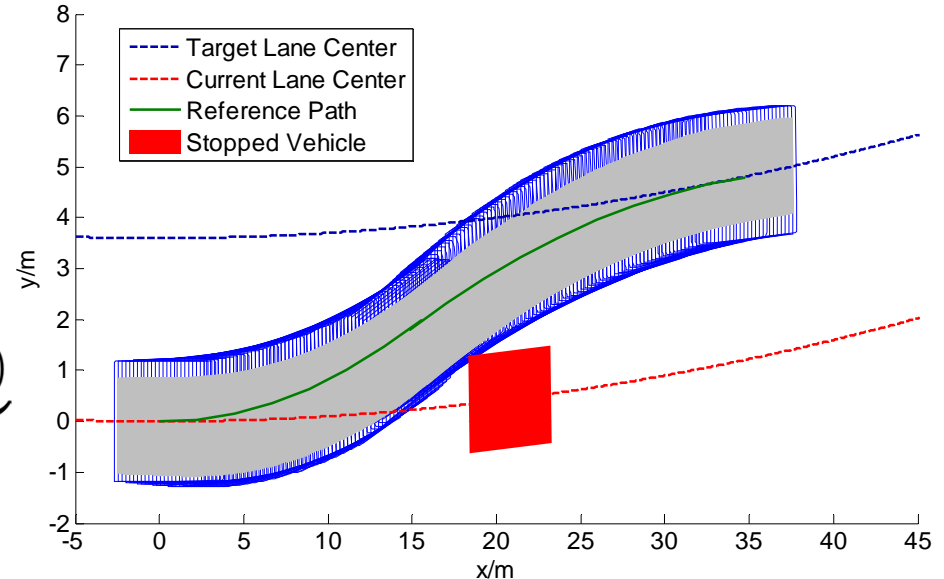
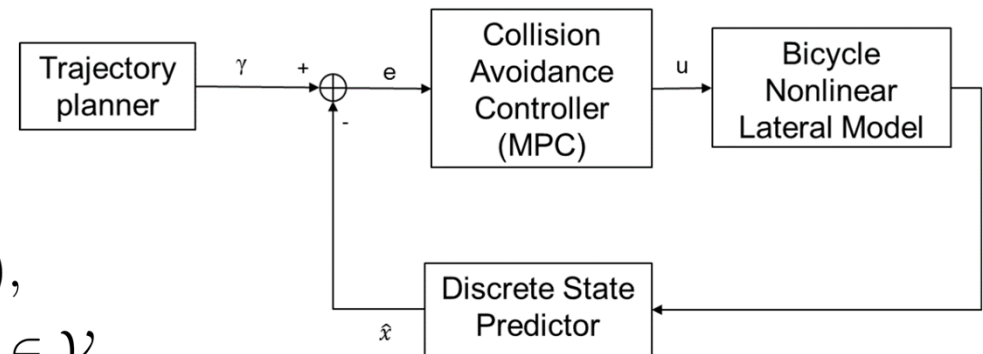
- Trajectory tracking controller based on MPC

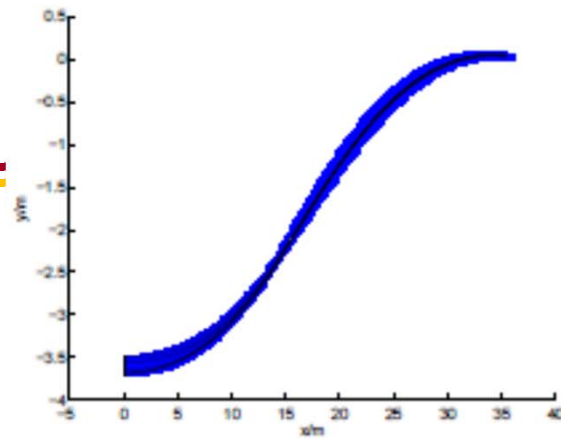
$$u(t) = -K(t)x(t) + h(\gamma(t), t)$$

- Overall linear system

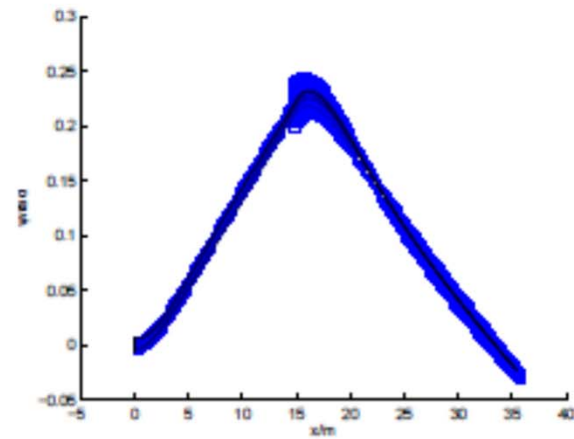
$$\dot{x}(t) = A_c(t)x(t) + \underbrace{v(t) + B(t)h(\gamma(t), t)}_{w(t) \in \mathcal{W}(t)}$$

- Verification result

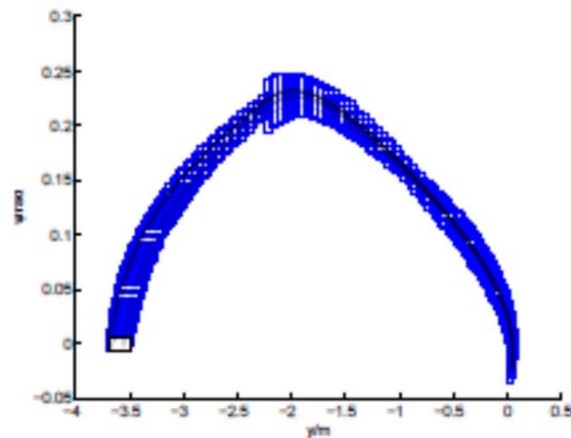




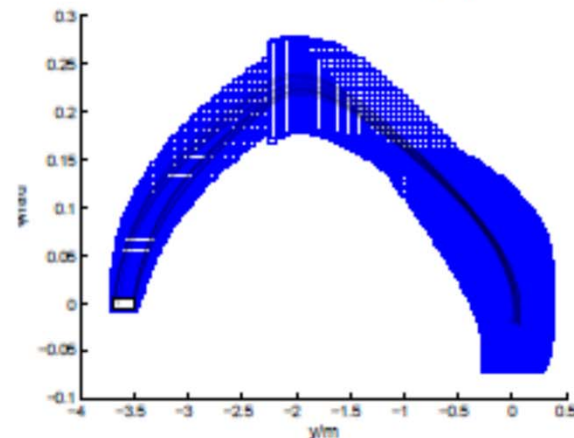
(a) Reachable set projected to longitude distance x and relative lateral distance y .



(b) Reachable set projected to longitude distance x and relative heading ψ .



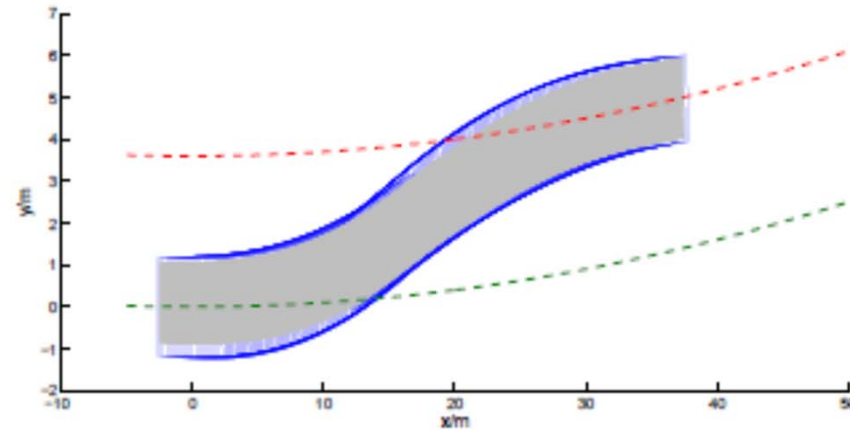
(c) Reachable set projected to relative lateral distance and relative heading.



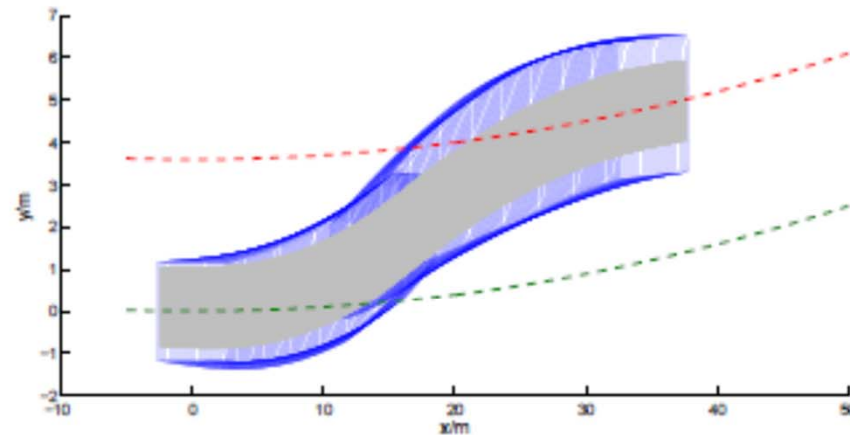
(d) Reachable set after adding deviation on lateral speed and yaw rate.

**Results
with
Noise**

Figure 2.4: Reachable set of linear hybrid system model. In all the plots, the reachable sets are shown as blue polytopes and simulation traces are black lines. The noise produced a large deviation on the reachable set (Fig. (c) and Fig. (d)). Although the system is still stable, the system is not robust in rejecting noise.



(a) Occupancy set of the vehicle for hybrid linear system model without sensor noise.



(b) Occupancy set of vehicle for hybrid linear system model with sensor noise

Figure 2.5: Clearly the hybrid linear model is not robust in terms of rejecting sensor noise, the occupancy set is enlarged to about 1m around the vehicle. In this particular case, the distance required to avoid collision is reduced for about 2m due to the sensor noise.

**Results
with
Noise**

Collaborative Coordination of UAS in the NAS with Safety Guarantees



- Introduction
- Existing Research
 - Dynamic Coverage Planning
 - Current Sensing Capabilities
 - Approaches to Low-Level Control
 - Approaches to Formal Safety Verification
- Framework and High-Level Dynamic Coverage
- Designing a Potential Function for Added Safety
 - Incorporating relative speed
- Safety Verification
 - Linear system
 - Nonlinear system with reference trajectory

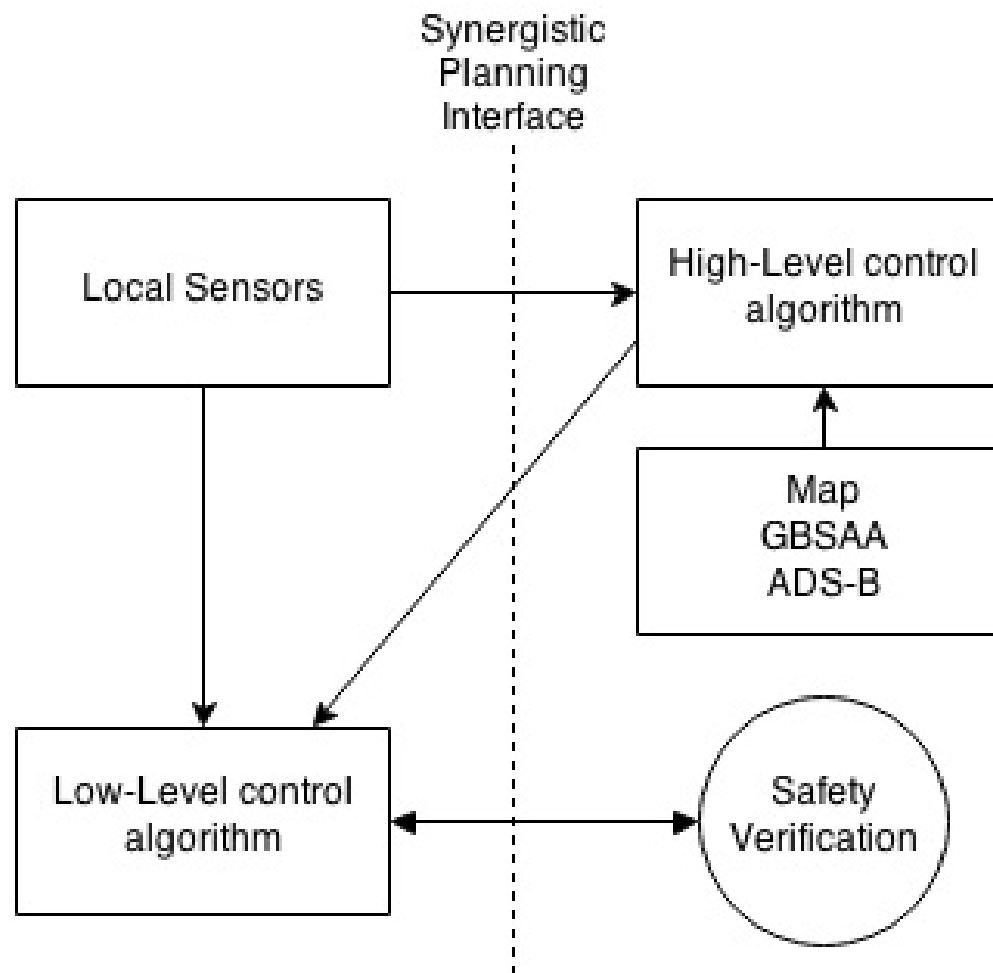
- Objectives
 - Coverage planning algorithm for multiple UAS agents
 - Collision avoidance of fast-moving piloted aircraft as well as other UAS agents
 - Utilization of a potential function to assure safety of the reference path
 - Online safety verification for individual agents
- Difficulties
 - High level plan vs low level safety verification.
 - Complex dynamics of quadrotor platform
 - Optimality vs safety

Existing Research

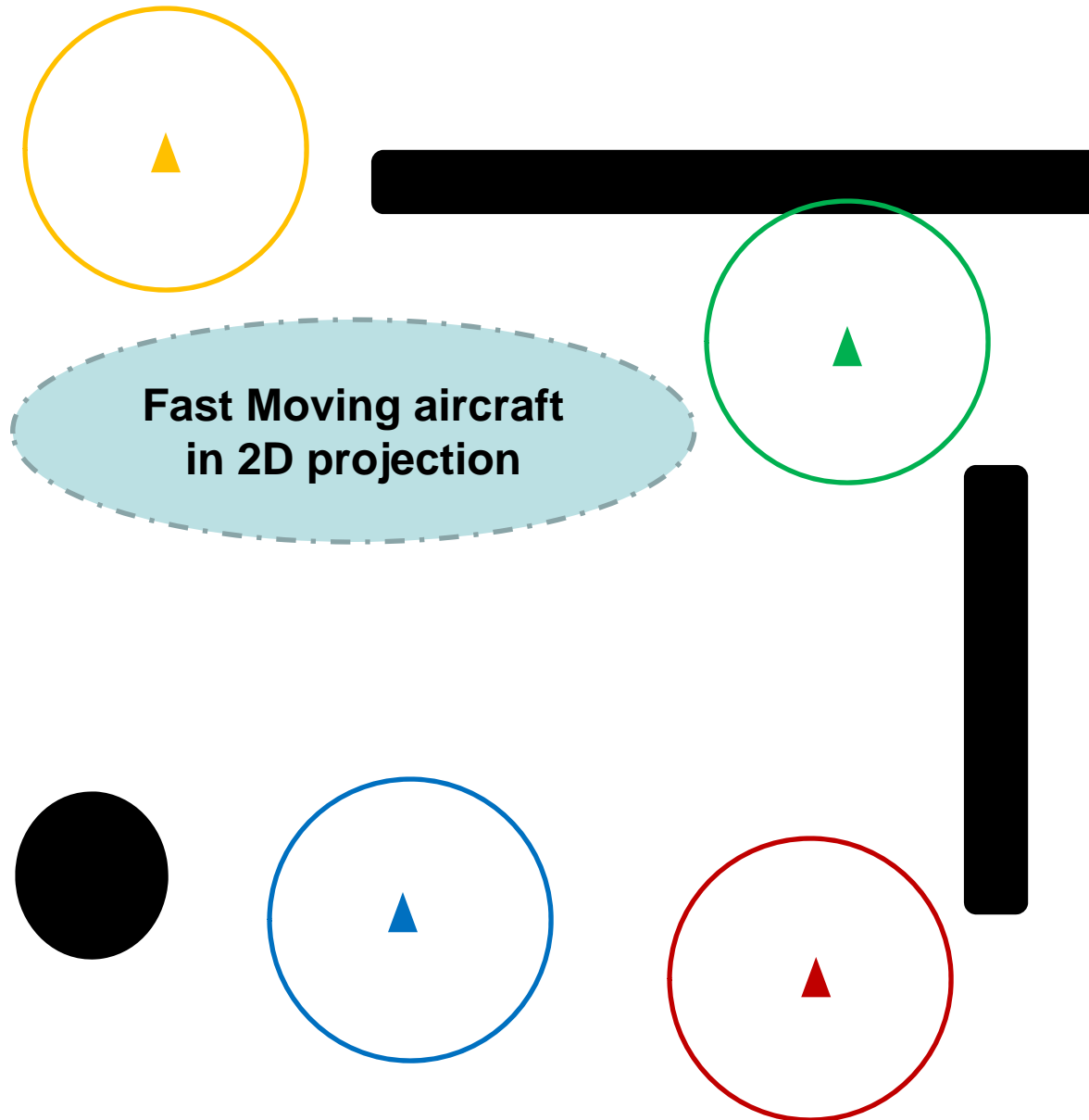
- Dynamic Sensor Coverage Planning (Heuristic and Otherwise)
 - Cellular decomposition and dynamic coverage for autonomous underwater vehicles [Hart 1998]
 - Approximate cell decomposition used by insects [Choset 2001]
 - Division of an area into flight corridors [Mellinger 2011]
 - Multi-resolution Cooperative Coverage [Gupta 2012]
- Existing Sensing Capabilities
 - GBSAA (Ground-Based Sense And Avoid) [Drake 2011]
 - Portable ADS-B Receiver [Garmin 2012]
 - Target tracking methods and turnkey systems [Parker 2002, Flir 2012]
- Approaches to Low-Level Control
 - Gradient Descent Algorithm using artificial potential fields [Hovareshti 2009, Khatib 1986]

- Approaches to Low-Level Control (continued)
 - Model Predictive Control for Quadrotor Platforms [Raffo 2010]
 - High Performance Waypoint Control for Quadrotor Platforms [Richter 2012]
- Approaches to Formal Safety Verification
 - Various methods for numerical analysis of reachable sets [Stursberg 2003, Asarin 2000, Guernic 2010, Althoff 2013]
 - Reachability for LTI [Girard 2006]
 - Reachability for Hybrid system with linear dynamics [Dang 2011]
 - Reachability for Nonlinear system using hybridization [Guernic 2009]
 - Reachability for nonlinear system using Taylor series expansion [Althoff 2010, 2011]
 - Application in Biology and Autonomous Vehicles [Dang 2011]

Framework



- Obstacles (black for static and blue for dynamic)
 - Static Obstacles
 - Physical obstacles such as buildings
 - UAS patrol area boundaries
 - Dynamic Obstacles
 - Manned aircraft moving through the patrol area.
 - Trajectories and positions are predicted with some uncertainty.



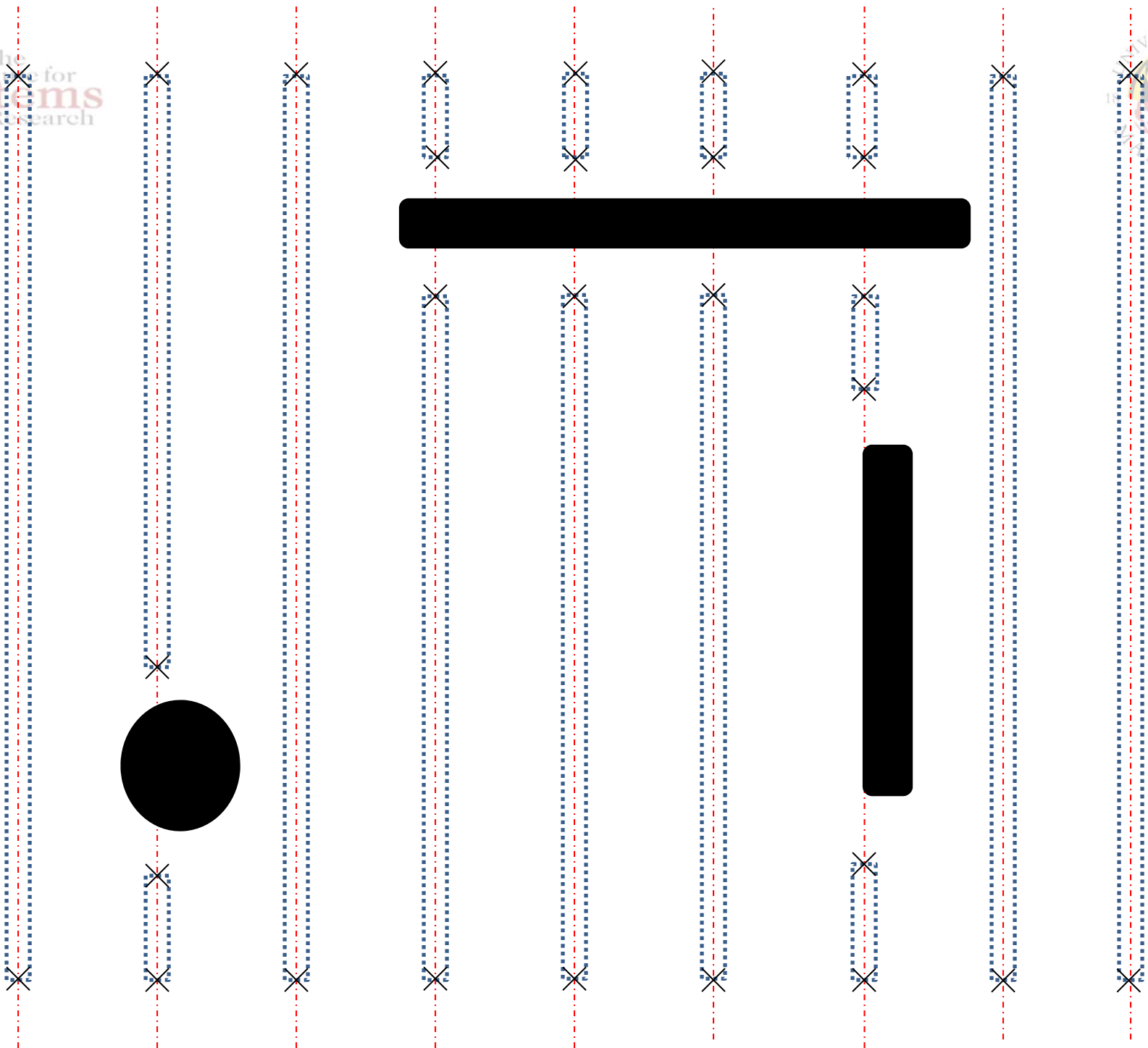
- Cell Decomposition (red dotted line)
 - Decompose the known target coverage area in an arbitrary direction (vertically in the illustration)
 - Cell width is defined by sensor range



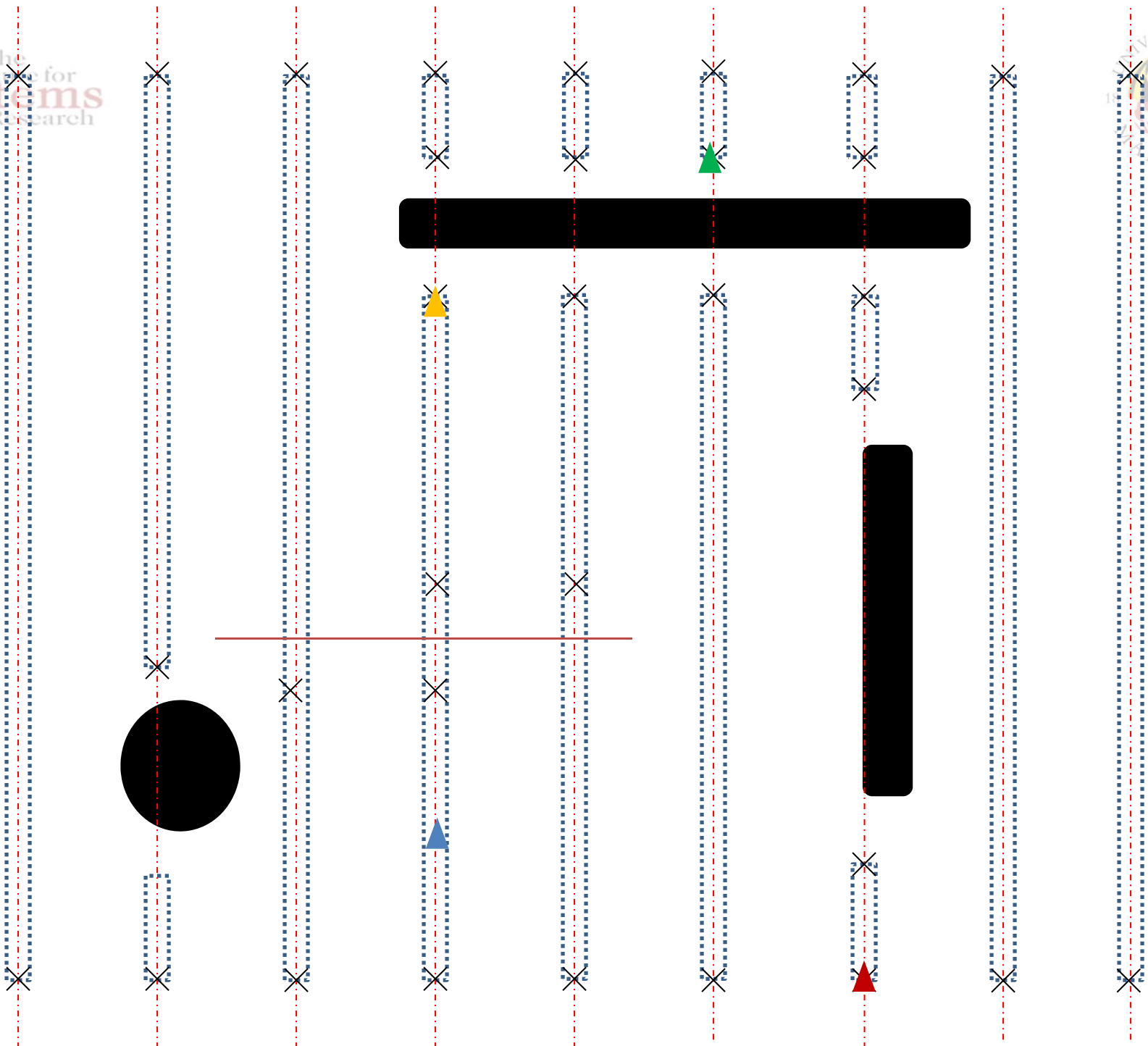
A diagram illustrating a fast-moving aircraft in 2D projection. The scene is defined by eight vertical red dashed lines. A light blue oval with a dashed border is positioned in the upper-middle section, containing the text "Fast Moving aircraft in 2D projection". Above this oval is a solid black horizontal rectangle. Below the oval is a solid black circle. To the right of the oval is a solid black vertical rectangle.

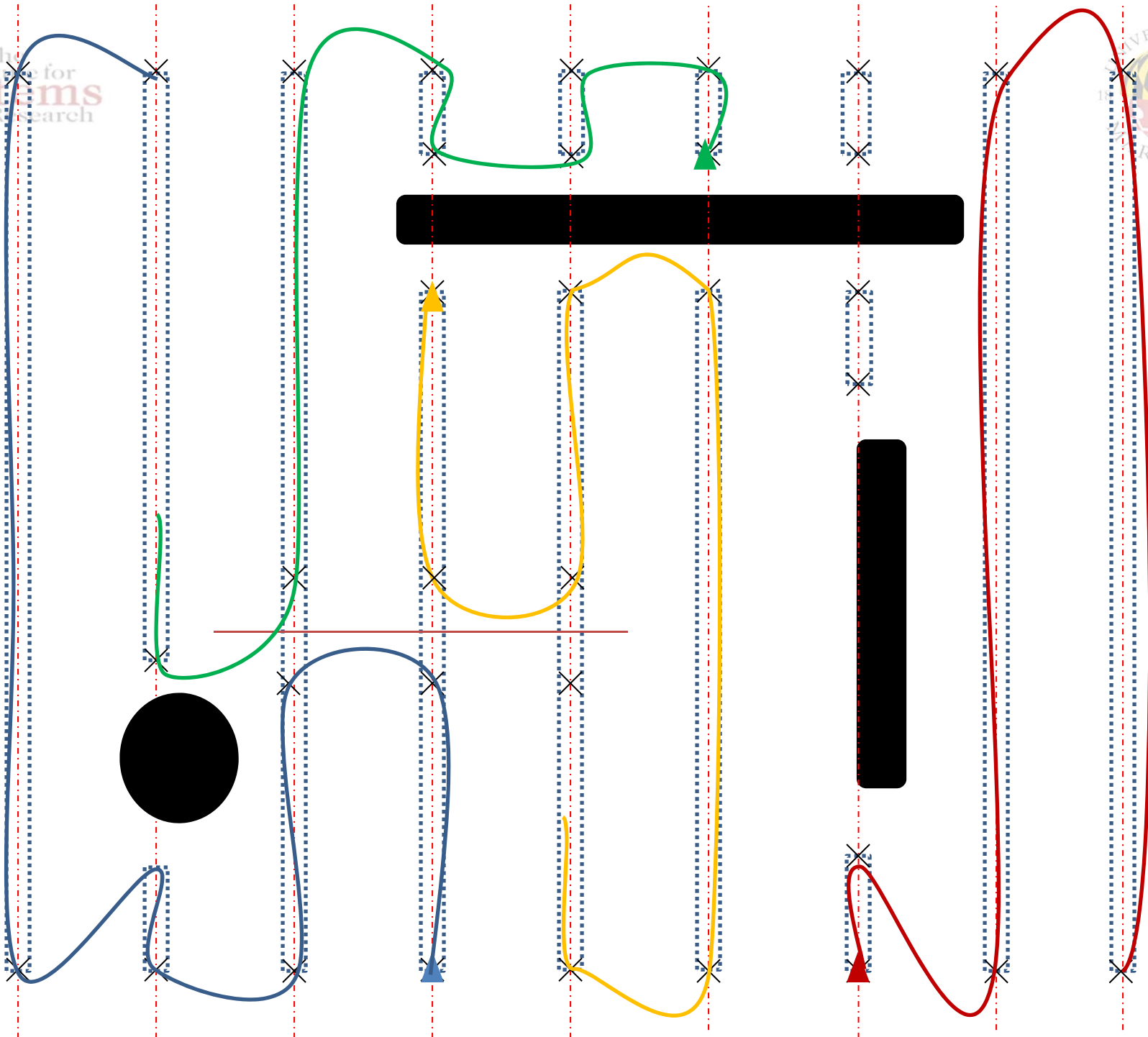
**Fast Moving aircraft
in 2D projection**

- Constraint Set (blue dotted box)
 - Assign virtual corridor constraint set along the direction of decomposition [Hert et al 1996, Mellinger 2011]
 - Sensor range and static obstacles define the two end points of the constraint sets
 - Within the corridor, UAS agent is in a patrol state, surveying and recording data in the region of interest.
- Waypoints (black x)
 - Ends of the corridors
 - Added when corridors are decomposed

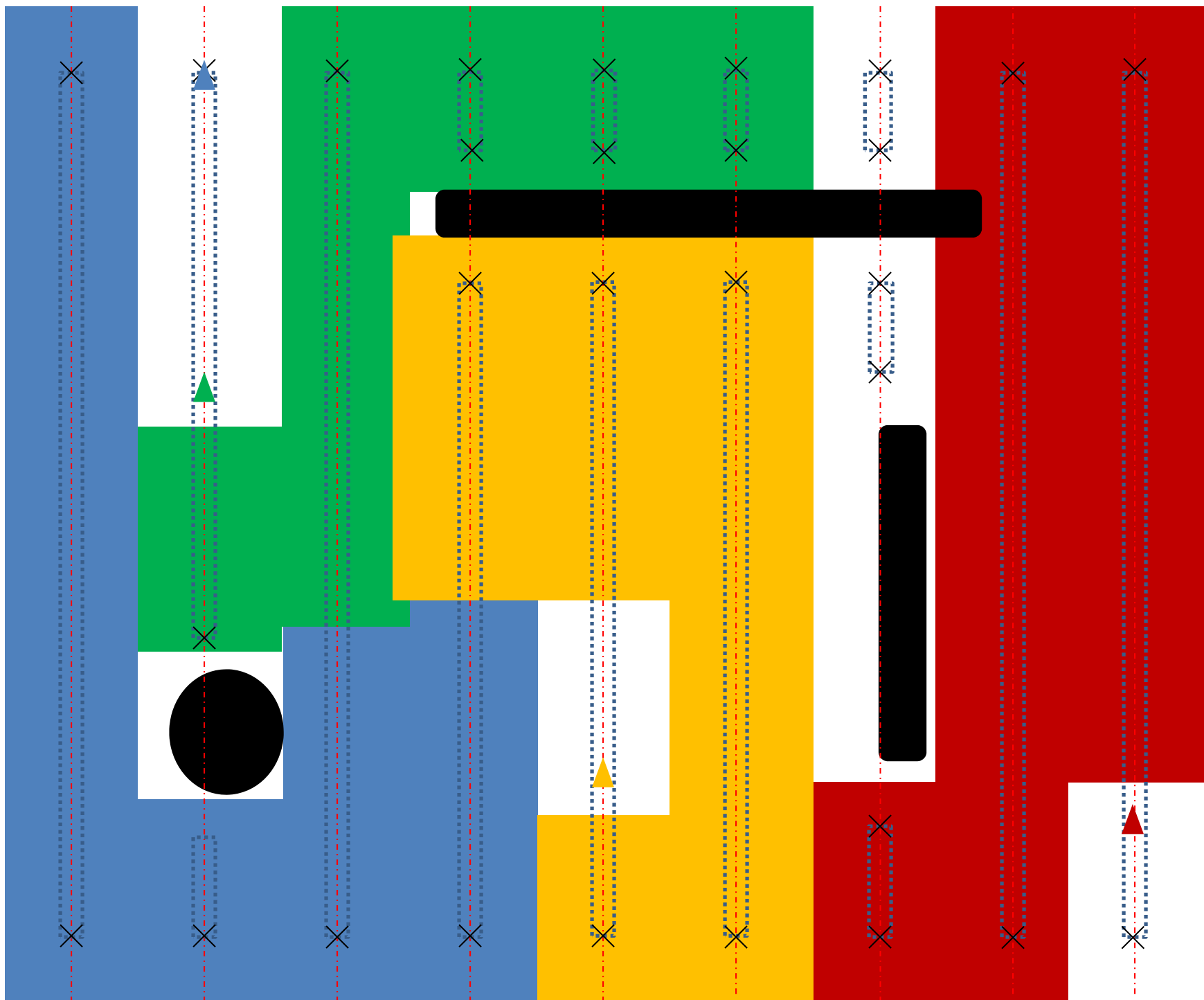


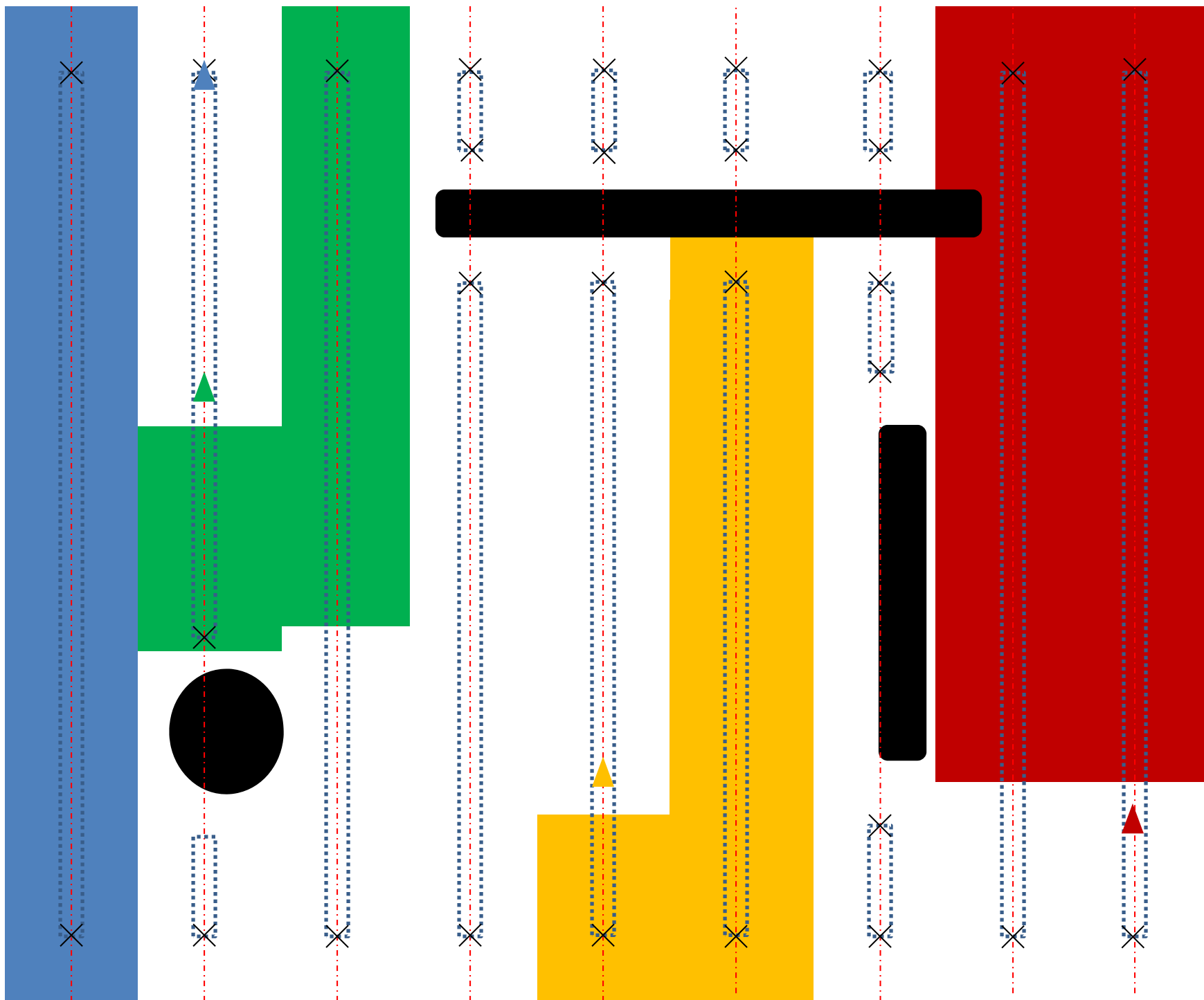
- Initial Location
 - Assign 4-5 agents to waypoints arbitrarily in space
- Corridor Decomposition
 - When two agents are at the same long corridor, decompose the shared corridor and its neighbor.
 - Agents will be assigned an adjacent corridor accordingly
 - If the corridor is short, one agent is assigned to move to other nearest corridor immediately
- Constraint Optimization
 - Corridor constraint
 - Moving to next waypoint





- Real Time Execution
 - Agents can meet in the same corridor. Decomposition is needed.
 - Coordinate to decide the next corridor to survey.
- Decay Time
 - Area being surveyed has a decay time. Over time the region will be required to be surveyed again. [Gupta 2012]
 - Waypoints will decay as the associated area decays. This will keep the complexity of the overall map stable.





Design of Potential Function for Added Safety

- Use a potential function to derive a local field based on velocity and avoid dynamic obstacles
 - Follow a desired trajectory assigned by the high-level planner
 - Repel away from other UAS agents and piloted aircraft
 - Incorporate relative speed in traffic avoidance

$$J_a(\mathbf{x}_a) = \sum_{i=1}^{N_a} b_i f(\mathbf{x}_a, \mathbf{x}_{ai}, \mathbf{v}_{ri})^{-1} + g(\mathbf{x}_a, \mathbf{x}_m, \mathbf{v}_m)^{-1} + \|\mathbf{x}_a - \mathbf{x}_{\gamma a}\|_2^2$$

$$\mathbf{x}_a \in \mathbb{R}^3 \setminus \mathcal{O}$$

$$J_a(\mathbf{x}_a) = \sum_{i=1}^{N_a} b_i f(\mathbf{x}_a, \mathbf{x}_{ai}, \mathbf{v}_{ri})^{-1} + g(\mathbf{x}_a, \mathbf{x}_m, \mathbf{v}_m)^{-1} + \|\mathbf{x}_a - \mathbf{x}_{\gamma a}\|_2^2$$

- Repel away from other UAS agents
 - f is similar to a squared Euclidean norm but modified to include relative velocity.

Agent position : \mathbf{x}_a

Position of i th agent : \mathbf{x}_{ai}

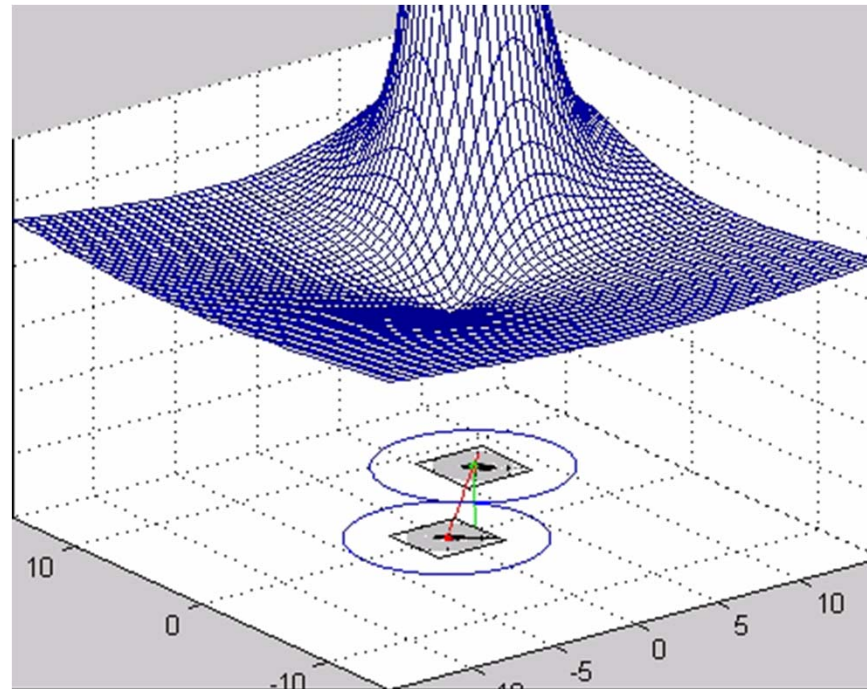
Relative velocity towards i th agent : \mathbf{v}_{ri}

$$f(\mathbf{x}_a, \mathbf{x}_{ai}, \mathbf{v}_{ri}) = (\mathbf{x}_a - \mathbf{x}_{ai})^T P_{ai} D_{ai} P_{ai}^T (\mathbf{x}_a - \mathbf{x}_{ai})$$

$$P_{ai} = \begin{pmatrix} \frac{v_{rix}}{\|\mathbf{v}_{ri}\|} \\ \frac{v_{riy}}{\|\mathbf{v}_{ri}\|} \\ \frac{v_{riz}}{\|\mathbf{v}_{ri}\|} \end{pmatrix} \quad \vec{o}_1 \quad \vec{o}_2 \quad D_{ai} = \begin{pmatrix} \|\mathbf{v}_{ri}\| & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad , \text{ where } \vec{o}_1, \vec{o}_2 \in \mathbb{R}^3 \text{ are picked to make the matrix } P_{ai} \text{ orthonormal.}$$

Potential Function

$$J_a(\mathbf{x}_a) = \sum_{i=1}^{N_a} b_i f(\mathbf{x}_a, \mathbf{x}_{ai}, \mathbf{v}_{ri})^{-1} + g(\mathbf{x}_a, \mathbf{x}_m, \mathbf{v}_m)^{-1} + \|\mathbf{x}_a - \mathbf{x}_{\gamma a}\|_2^2$$



UAS agents avoid one another.

Potential Function

$$J_a(\mathbf{x}_a) = \sum_{i=1}^{N_a} b_i f(\mathbf{x}_a, \mathbf{x}_{ai}, \mathbf{v}_{ri})^{-1} + g(\mathbf{x}_a, \mathbf{x}_m, \mathbf{v}_m)^{-1} + \|\mathbf{x}_a - \mathbf{x}_{\gamma a}\|_2^2$$

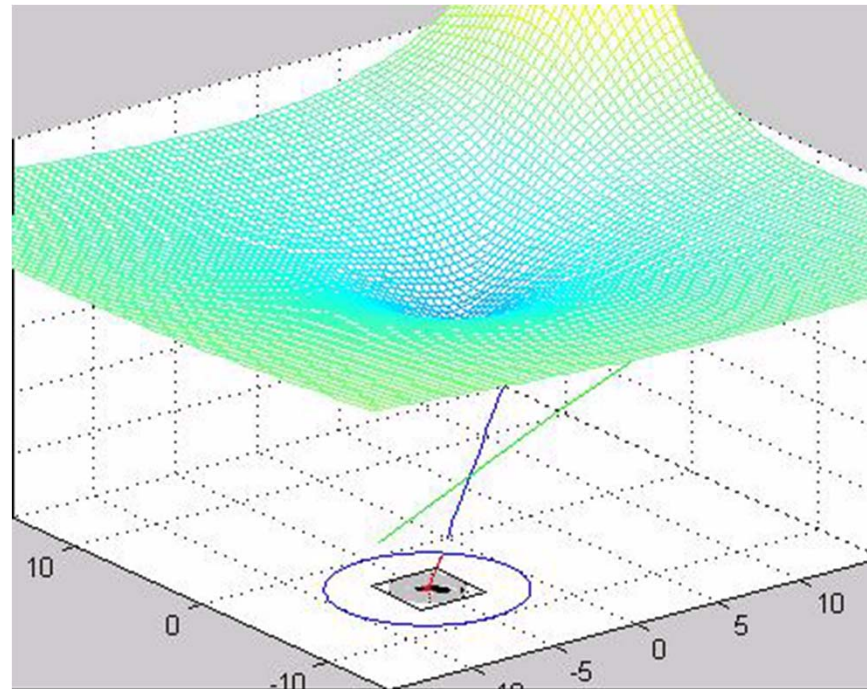
- Repel away from piloted aircraft

$$g(\mathbf{x}_a, \mathbf{x}_m, \mathbf{v}_m)^{-1} = \left((\mathbf{x}_a - \mathbf{x}_m)^T P_m D_m P_m^T (\mathbf{x}_a - \mathbf{x}_m) \right)^{-1}$$

- P_m and D_m are constructed similarly to P_{ai} and D_{ai}
- g is designed to create a steep gradient
- Strongest potential gradient is used to push UAS agents out of the way of any piloted aircraft
- Radar-based GBSAA might provide locations of piloted aircraft

Potential Function

$$J_a(\mathbf{x}_a) = \sum_{i=1}^{N_a} b_i f(\mathbf{x}_a, \mathbf{x}_{ai}, \mathbf{v}_{ri})^{-1} + g(\mathbf{x}_a, \mathbf{x}_m, \mathbf{v}_m)^{-1} + \|\mathbf{x}_a - \mathbf{x}_{\gamma a}\|_2^2$$



Gradient pushes UAS out of the way of piloted aircraft.

Potential Function

$$J_a(\mathbf{x}_a) = \sum_{i=1}^{N_a} b_i f(\mathbf{x}_a, \mathbf{x}_{ai}, \mathbf{v}_{ri})^{-1} + g(\mathbf{x}_a, \mathbf{x}_m, \mathbf{v}_m)^{-1} + \|\mathbf{x}_a - \mathbf{x}_{\gamma a}\|_2^2$$

- Follow desired trajectory from the high level planner and optimized trajectory.
 - Each agent is “attracted” to its reference trajectory by a potential field.
 - If the agent is delayed, the target point along its trajectory will stop and wait.
 - In the quadrotor example, the optimal trajectory can be modified to minimize snap (second derivative of acceleration).

Safety Verification: Reachability analysis for LTI

- LTI System, $B = I$

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + \mathbf{u}(t) \quad \mathbf{x}, \mathbf{u} \in \mathbb{R}^n, \mathbf{x}(0) = \mathbf{x}_0$$

- Separation

$$\mathbf{x}(t) = e^{tA} \mathbf{x}_0 + \int_0^t e^{(t-\tau)A} \mathbf{u}(\tau) d\tau$$

- Reachable Set for LTI

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + \mathbf{u}(t), \quad \mathbf{x}(0) \in \mathcal{X}_0, \mathbf{u}(t) \in \mathcal{U}$$

$$\mathcal{R}_\delta(\mathcal{X}_0) = e^{\delta A} \mathcal{X}_0 \oplus \mathcal{R}_\delta(\{0\})$$

Reachability Analysis for the Nonlinear System

- Linearization

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}, \mathbf{u}, \mathbf{x}_d)$$

$$= f(\mathbf{x}^*, \mathbf{u}^*, \mathbf{x}_d) + \nabla_{\mathbf{x}} f \Big|_{\mathbf{x}=\mathbf{x}^*} (\mathbf{x} - \mathbf{x}^*)(t) + \nabla_{\mathbf{u}} f \Big|_{\mathbf{u}=\mathbf{u}^*} (\mathbf{u} - \mathbf{u}^*)(t)$$

+ higher order term,

$$\mathbf{x}(0) \in \mathcal{X}_0, \mathbf{u}(t) \in \mathcal{U}$$

- Separation

- Enlarge by inputs
- Enlarge by linearization error

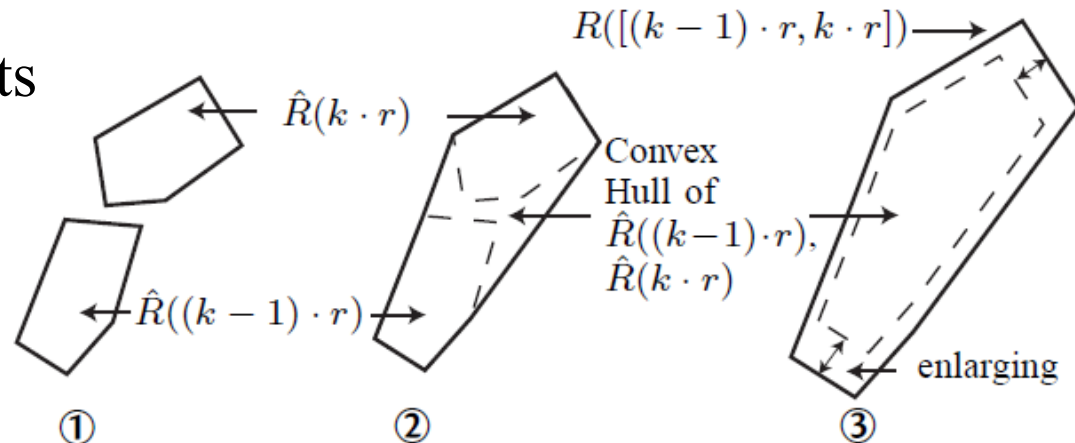


Image courtesy of [Althoff 2010]: Safety Verification of Autonomous Vehicles for Coordinated Evasive Maneuvers
Matthias Althoff, Daniel Althoff, Dirk Wollherr and Martin Buss

Nonlinear System with Reference Trajectory

- Given a reference trajectory, is the autonomous vehicle or aircraft safe?
 - Trajectory separation \neq safe
 - Modeling error, uncaptured dynamics and initial state uncertainty
 - Lyapunov stability analysis is not enough
 - Reference trajectory is updated online
 - Aerodynamic interaction and modeling uncertainty
- Linearization with reference trajectory

Let $\mathbf{z} = [\mathbf{x}, \mathbf{u}]$,

then the inclusion can be formally described as

$$\dot{\mathbf{x}}(t) \in f(\mathbf{z}^*, \mathbf{x}_d) + \nabla_{\mathbf{z}} f \Big|_{\mathbf{z}=\mathbf{z}^*} (\mathbf{z} - \mathbf{z}^*)(t) + \frac{1}{2} (\mathbf{z} - \mathbf{z}^*)^T \nabla_{\mathbf{z}}^2 f \Big|_{\mathbf{z}=\xi} (\mathbf{z} - \mathbf{z}^*)(t)$$

where ξ takes values in the set $\{\mathbf{z}^* + \alpha(\mathbf{z} - \mathbf{z}^*) | \alpha \in [0, 1]\}$

Quadrotor Platform Dynamics

- State variables
 - Position $\mathbf{p} = (p_x, p_y, p_z)$
 - Position derivatives $\mathbf{v} = (v_x, v_y, v_z)$
 - Attitude (roll pitch yaw) $\mathbf{R} = (\phi, \theta, \psi)$
 - Body frame rotational velocities $\mathbf{\Omega} = (p, q, r)$
 - $\mathbf{x} = (p_x, p_y, p_z, v_x, v_y, v_z, \phi, \theta, \psi, p, q, r)$
- Inputs
 - Total thrust F aligned with body frame “up”
 - Input variable $\mathbf{u} = (F, M_1, M_2, M_3)$
 - Rotational momentum $\mathbf{M} = (M_1, M_2, M_3)$

Quadrotor Platform Dynamics

- Dynamics

$$\begin{cases} \dot{\mathbf{p}} = \mathbf{v} \\ \dot{\mathbf{v}} = -g\mathbf{e}_3 + \frac{F}{m}R\mathbf{e}_3 \\ \dot{\mathbf{R}} = R_w^{-1}\boldsymbol{\Omega} \\ \dot{\boldsymbol{\Omega}} = J^{-1}(\mathbf{M} - \boldsymbol{\Omega} \times J\boldsymbol{\Omega}) \end{cases} \quad R_w = \begin{pmatrix} c\theta & 0 & c\phi s\theta \\ 0 & 1 & s\phi \\ s\theta & 0 & c\phi c\theta \end{pmatrix}$$

$$R = \begin{pmatrix} c\theta c\psi - s\theta s\phi s\psi & -c\phi s\psi & c\psi s\theta + c\theta s\phi s\psi \\ c\psi s\theta s\phi + c\theta s\psi & c\phi c\psi & -c\theta c\psi s\phi + s\theta s\psi \\ -c\phi s\theta & s\phi & c\theta c\phi \end{pmatrix}$$

Quadrotor Trajectory Following Controller

- Thrust Controller

$$\mathbf{e}_p = \mathbf{p}_d - \mathbf{p}, \mathbf{e}_v = \mathbf{v}_d - \mathbf{v}$$

$$\mathbf{F}_{des} = K_p \mathbf{e}_p + K_v \mathbf{e}_v + m g \mathbf{e}_3 + m \mathbf{a}_d$$

$$F = \mathbf{F}_{des} \cdot \mathbf{z}_B$$

- Attitude Controller

$$\mathbf{e}_R = 1 / 2 (-R_d^T R + R^T R_d)^\vee, \mathbf{e}_w = w_d - w$$

$$\mathbf{M} = K_R \mathbf{e}_R + K_w \mathbf{e}_w$$

– $^\vee$ is the *vee mapping* from $\text{SO}(3)$ to \mathbb{R}^3

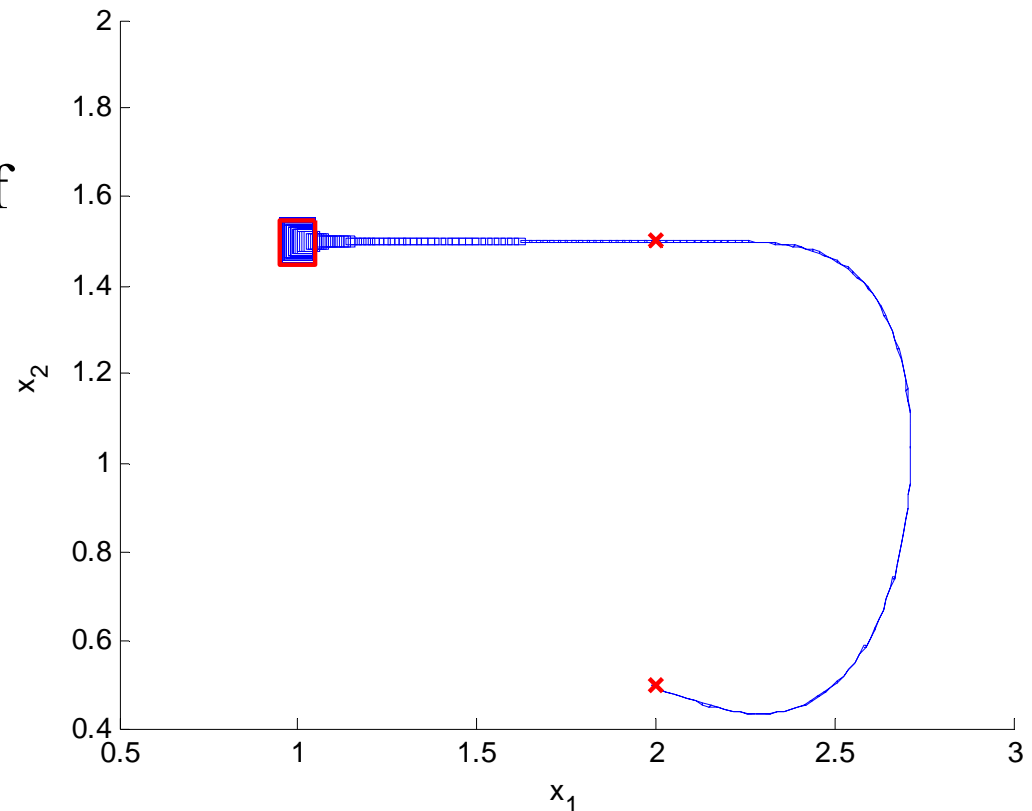
- Overall differential equations

$$\begin{aligned} \dot{\mathbf{x}}(t) \in & f(\mathbf{z}^*, \mathbf{x}_d) + \nabla_{\mathbf{z}} f \Big|_{\mathbf{z}=\mathbf{z}^*} (\mathbf{z} - \mathbf{z}^*)(t) \\ & + \frac{1}{2} (\mathbf{z} - \mathbf{z}^*)^T \nabla_{\mathbf{z}}^2 f \Big|_{\mathbf{z}=\xi} (\mathbf{z} - \mathbf{z}^*)(t), \end{aligned}$$

$$\mathbf{x}(0) \in \mathcal{X}_0, \mathbf{u}(t) \in \mathcal{U}$$

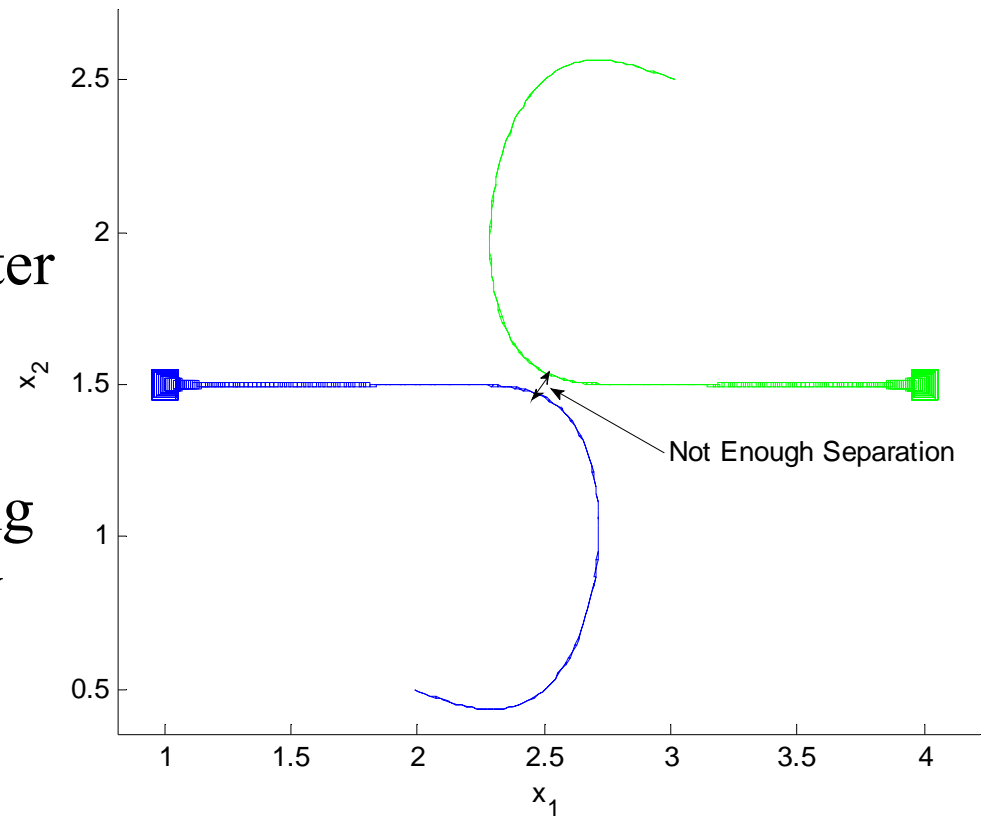
Quadrotor Safety Analysis

- Reachable set computation for agile maneuver of one quadrotor in xy projection.
- The initial set has 0.05m of on position uncertainty. Limited disturbance is added.
- Starting set is highlighted by red square. Waypoints are marked as red 'x'
- It can be seen that the system without disturbance is fairly stable.



Quadrotor Safety Analysis

- Combining two UAV reachable set together, we conclude that the original plan is not safe, since the separation between the center of the UAV is not large enough.
- Future work involves adding disturbance and uncertainty to different state of the system based on physical platform and environment.



Ongoing Work

- Add a variety of UAS states
 - Takeoff
 - Land
 - Emergency
- Integrate simulations with other simulations of NAS traffic

Part I.2: Control Synthesis for Collision Avoidance

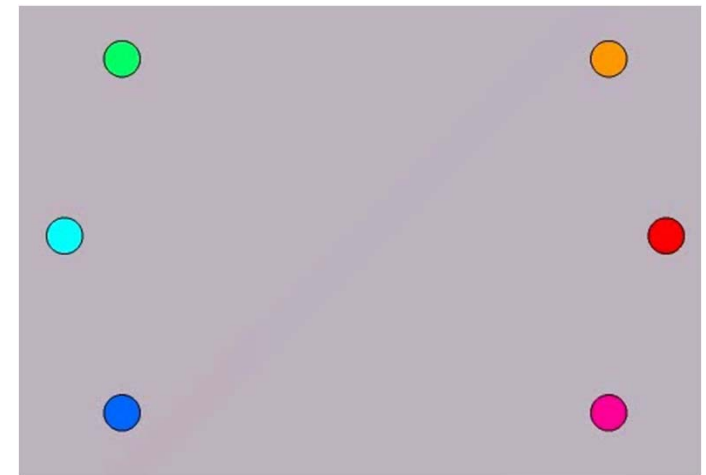
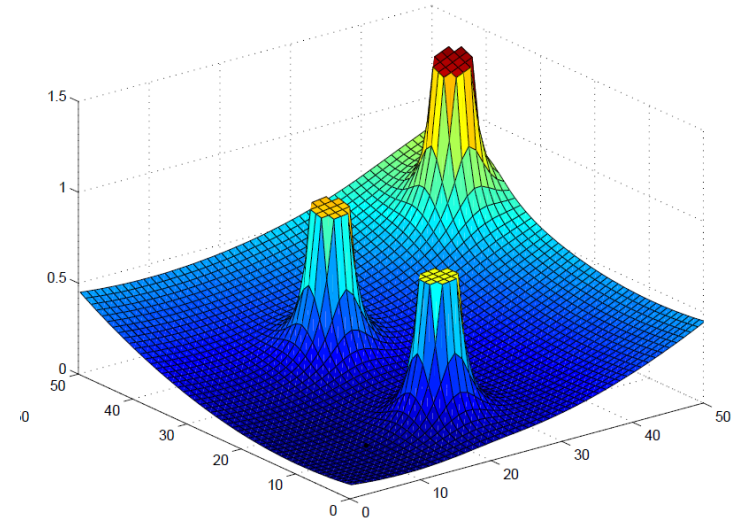
- Control **synthesis** of safe reachable tubes for collision avoidance using **convex optimization**
 - Proposed a method to convert the collision avoidance of reachable tubes to convex optimization problems
 - Analyzed for **collaborative** and non-collaborative settings
 - Resulting **control tube** can be constant over time² or time varying³
 - Demonstrated on high dimensional **quadrotor** and **fixed-wing** dynamic model



2. Y. Zhou and J. S. Baras, "Reachable Set Approach to Collision Avoidance for UAVs", Proceedings 54th IEEE Conference on Decision and Control, Osaka, Japan, pp. 5947-5952, December 15-18, 2015.
3. Y. Zhou, A. Raghavan and J. S. Baras, "Time Varying Control Set Design for UAV Collision Avoidance Using Reachable Tubes", Proceedings 55th IEEE Conference on Decision and Control, Las Vegas, USA, pp. 6857-6862, December 12-14, 2016.

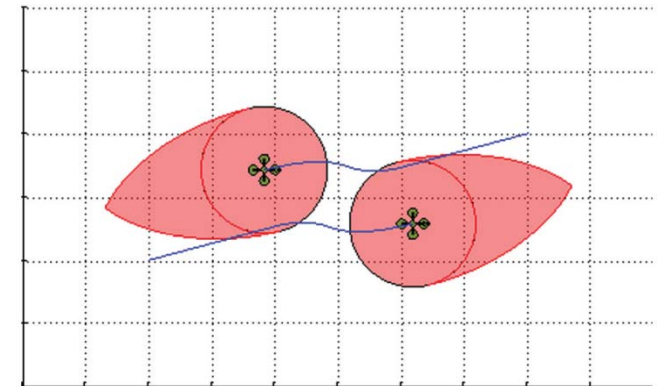
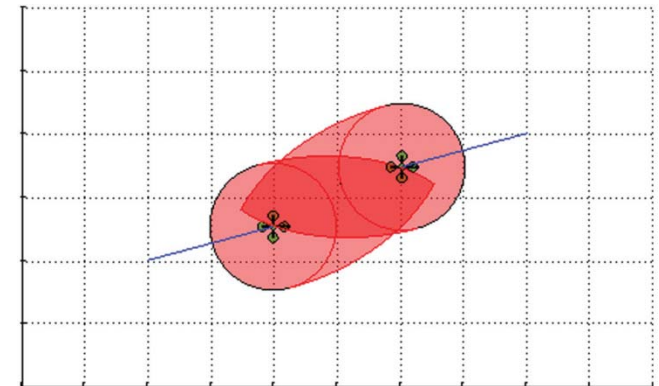
State of Art: Collision Avoidance

- Existing approaches for trajectory based collision avoidance
 - Artificial potential function based planning
 - Velocity obstacle based avoidance
 - Rule based
 - Re-plan based



State of the Art and Challenges

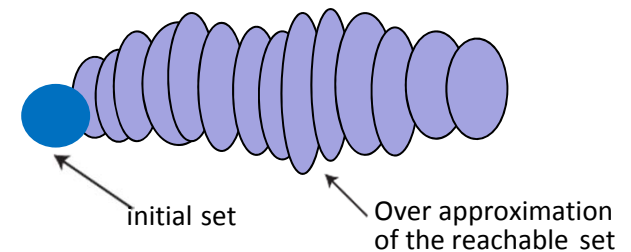
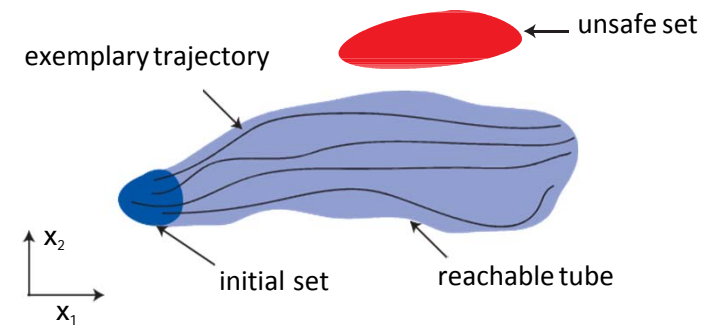
- Existing approaches for trajectory based collision avoidance based on tubes⁴
 - Artificial potential function based planning [N. E. Leonard and E. Fiorelli 2001]
 - Collision cone based [C. Carbone 2006]
 - Based on tubes⁴. Game theoretic framework, treat other vehicles as adversaries. Nonlinear dynamics. Complex, requires solving PDEs, does not scale to high dimensional space
- Challenges
 - How to provide **safety guarantees** for **unknown execution** of the other vehicles
 - How to avoid other vehicles given sensor uncertainties and control disturbances in a **collaborative** setting (real-time and online)



4. Gillula, J. H., Hoffmann, G. M., Huang, H., Vitus, M. P., & Tomlin, C. "Applications of hybrid reachability analysis to robotic aerial vehicles." *The International Journal of Robotics Research*, 0278364910387173, 2011

Reachable Set Based Safety Analysis

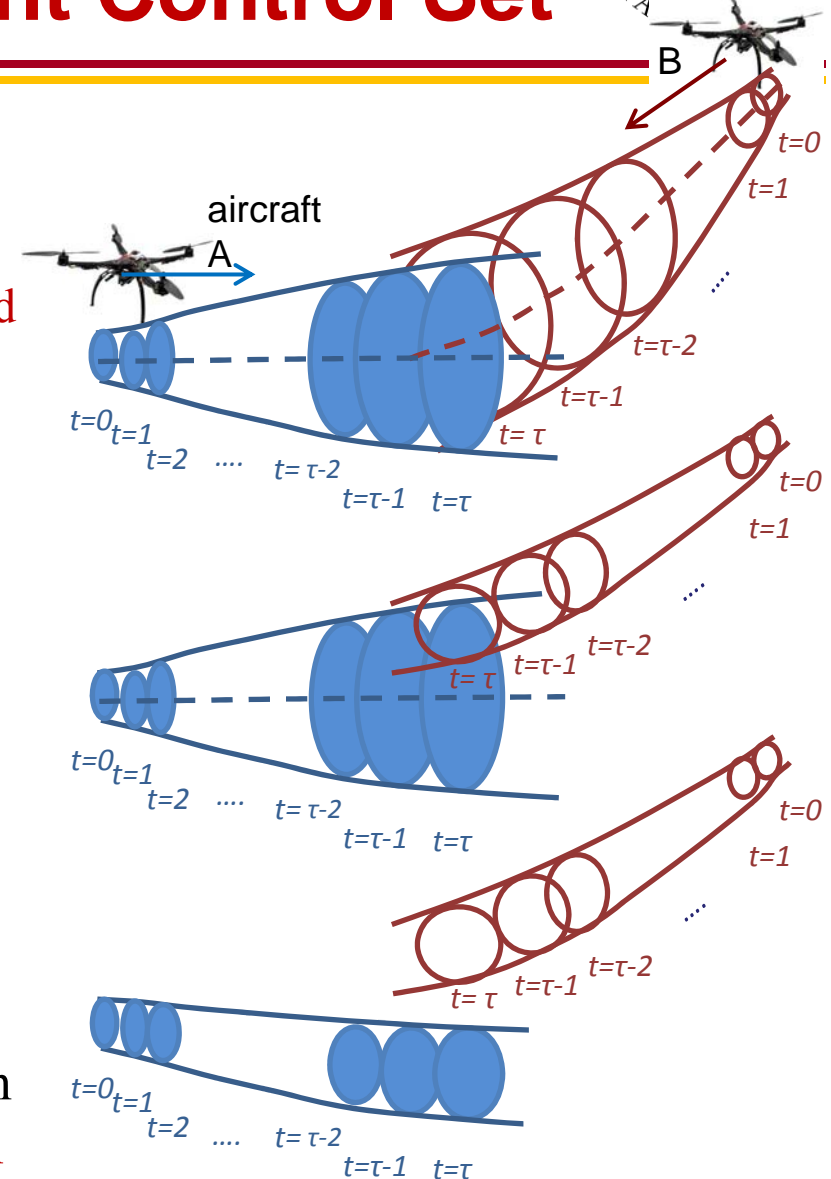
- Reachable set $\mathcal{R}[\theta]$ of a dynamics at time θ is defined as set of states reachable from a **bounded initial set** X_0 , a **control set** U and a **disturbance set** V .
- The control sets are **synthesized** so that the reachable sets of the UAVs have no intersection.
- Efficient reachable set computation normally uses convex over-approximations such as **ellipsoids** and polytopes and **linearized** dynamics



Collision Avoidance of Two UAVs with Constant Control Set



- Goal: seek a control set design for aircraft A and B
 - The new **control constraint sets** are more **restricted** than their initial ones.
 - Guarantee collision avoidance **between reachable sets**
- Decompose the problem to two parts
 - Seek a tighter control constraint set for aircraft B such that
 - **Collision free** from nominal trajectory of aircraft A
 - The control set of B remains large enough.
 - Seek a safe reachable tube for aircraft A such that A and B are apart by required **separation**



Linear Dynamics and Convex Sets

- For Linear Dynamics given by

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) + v(t), \quad (2)$$

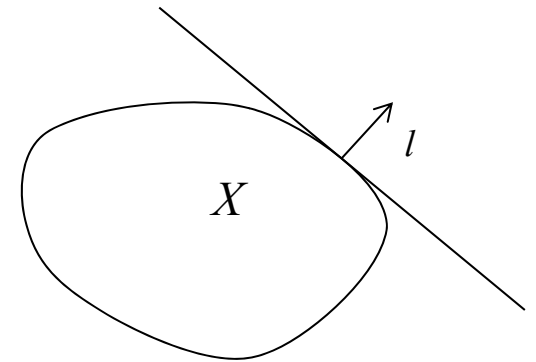
where $\mathcal{U}(t)$, \mathcal{X}_0 and \mathcal{V} are all convex and compact sets.

- Lemma:

With $\mathcal{U}(t)$, \mathcal{X}_0 and \mathcal{V} being convex and compact, the reachable set $\mathcal{R}[\vartheta]$ is also convex and compact.

- Support Function

Denote $\rho(l|X)$ to be the support function of the set X at direction l . $\rho(l|X) = \max\{\langle l, x \rangle \mid x \in X\}$



- Nonlinear dynamics can be linearized around nominal controls $u^*(t)$ and trajectory $x^*(t)$

$$A(t) = \left. \frac{\partial f}{\partial x} \right|_{x=x^*}, \quad B(t) = \left. \frac{\partial f}{\partial u} \right|_{u=u^*}$$

Ellipsoids and the Associated Reachable Set

- If we assume all the sets are approximated in terms of ellipsoids

- Let c_X and M_X denote the center and shape matrix of an ellipsoid, then if $x \in X = E(c_X, M_X)$

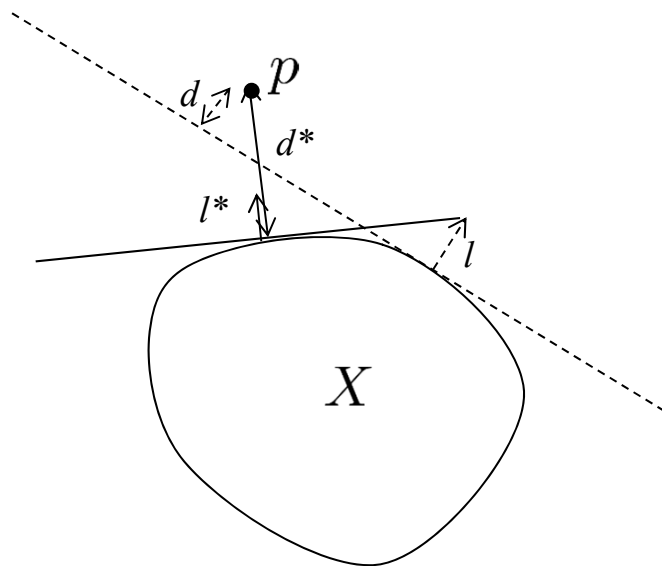
$$\langle x - c_X, M_X^{-1}(x - c_X) \rangle \leq 1.$$

- The Reachable Set can then be represented by the following,

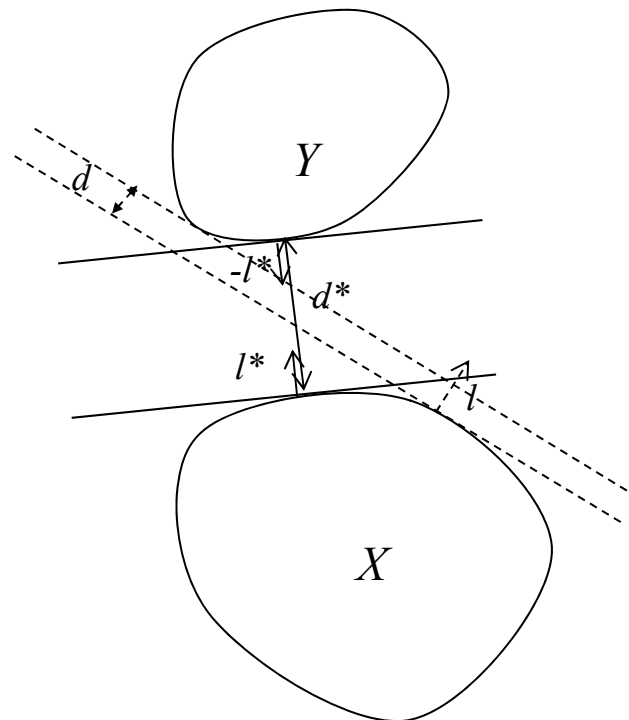
$$\begin{aligned} \rho(l|\mathcal{R}[\vartheta]) = & \langle l, \Phi(\vartheta, 0)c_{X_0} \rangle + \left\langle l, \int_0^\vartheta \Phi(\vartheta, s)B(s)c_{\mathcal{U}}(s)ds \right\rangle \\ & + \langle l, \Phi(\vartheta, 0)M_{X_0}\Phi^T(\vartheta, 0)l \rangle^{1/2} + \left\langle l, \int_0^\vartheta \Phi(\vartheta, s)c_{\mathcal{V}}(s)ds \right\rangle \\ & + \int_0^\vartheta \langle l, \Phi(\vartheta, s)B(s)M_{\mathcal{U}}B^T(s)\Phi^T(\vartheta, s)l \rangle^{1/2} ds \\ & + \int_0^\vartheta \langle l, \Phi(\vartheta, s)M_{\mathcal{V}}\Phi^T(\vartheta, s)l \rangle^{1/2} ds \end{aligned}$$

Distance Using Support Function

$$\text{dist}(p, X) = \max_l \langle l, p \rangle - \rho(l|X)$$



$$\text{dist}(X, Y) = \max_l -\rho(-l|Y) - \rho(l|X)$$



- Assume that we know that the time to collision between the UAVs is τ s, the **distance** between the **reachable set of B** and **estimated nominal trajectory of aircraft A** c_X^A is:

$$\max_l \langle l, c_X^A(\tau) \rangle - \rho(l | \mathcal{R}_x^B(\tau))$$

- The design parameters are the **center and shape matrix of the new control set** denoted as $E(q(t), Q(t)^T Q(t))$.
- The first optimization problem can then be described by

$$\begin{aligned} & \max_{q(t), Q(t)} \max_l \quad \langle l, c_X^A(\tau) \rangle - \rho(l | \mathcal{R}_x^B(\tau)) + k \int_0^\tau \log(\det(Q(t))) dt \\ & \text{subject to} \quad E(q(t), Q(t)^T Q(t)) \subset \mathcal{U}^B \quad \forall t \in [0, \tau] \\ & \quad \quad \quad \|l\| = 1 \\ & \quad \quad \quad \langle l, c_X^A(\tau) \rangle - \rho(l | \mathcal{R}_x^B(\tau)) \geq 0 \end{aligned}$$

- To simplify, we fix the separation direction l and remove the time dependency of q , and Q .

$$\max_{q \in \mathbb{R}^m, Q \in \mathbb{R}^{m \times m}} \langle l^*, c_X^A(\tau) \rangle - \rho(l^* | \mathcal{R}_x^B(\tau)) + k(\log \det(Q))$$

subject to

$$\lambda > 0$$

$$\begin{bmatrix} 1 - \lambda & 0 & (q - c_U^B)^T \\ 0 & \lambda I & Q \\ q - c_U^B & Q & M_U^B \end{bmatrix} \succeq 0$$

$$\langle l^*, c_X^A(\tau) \rangle - \rho(l^* | \mathcal{R}_x^B(\tau)) \geq 0$$

$$\begin{aligned} \rho(l | \mathcal{R}_x^B(t)) &= \langle l, e^{At} c_{X_0}^B \rangle + \left\langle l, \int_0^t e^{A(t-s)} ds B q \right\rangle + \langle l, e^{At} M_{X_0}^B e^{At} l \rangle^{1/2} \\ &+ \int_0^t \left\langle l, e^{A(t-s)} B Q^T Q B^T (e^{A(t-s)})^T l \right\rangle^{1/2} ds. \end{aligned}$$

- $\rho(l|\mathcal{R}_x^B(\tau))$ can be upper bounded by $\tilde{\rho}(l|\mathcal{R}_x^B(\tau))$ using the **matrix norm property**. Then the objective function can be **lower bounded** by $\langle l^*, c_X^A(\tau) \rangle - \tilde{\rho}(l^*|\mathcal{R}_x^B(\tau))$ where

$$\begin{aligned} \tilde{\rho}(l|\mathcal{R}_x^B(t)) = & \langle l, e^{At} c_{X_0}^B \rangle + \left\langle l, \int_0^t e^{A(t-s)} ds B q \right\rangle + \langle l, e^{At} M_{X_0}^B e^{At} l \rangle^{1/2} \\ & + \|Q\|_2 \int_0^t \left\langle l, e^{A(t-s)} B B^T (e^{A(t-s)})^T l \right\rangle^{1/2} ds. \end{aligned}$$

- This makes the whole problem **convex**
- We then used CVX⁵ to solve the convex optimization problem

- After the reachable set of aircraft B is determined, we compute the **largest** control set of aircraft A such that the reachable set **maintain enough separation**.

$$\begin{aligned} & \max_{q \in \mathbb{R}^m, Q \in \mathbb{R}^{m \times m}} \log(\det(Q)) \\ & \text{subject to} \quad E(q, Q^T Q) \subset \mathcal{U}^A \\ & \quad -\rho(-l^* | \mathcal{R}_x^A(\tau)) - \rho(l^* | E(c_X^B, M_X^B)) > 0 \end{aligned}$$

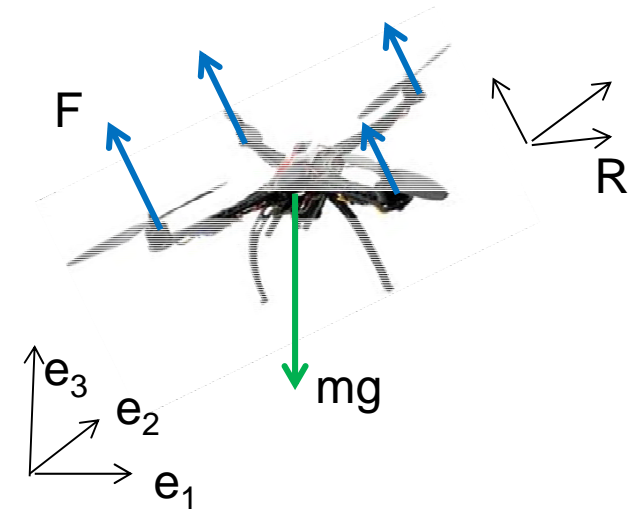
- The constraint can be again relaxed to obtain a **convex** one.

$$\begin{aligned} & \langle l^*, e^{At} c_{X_0}^A \rangle + \left\langle l^*, \int_0^t e^{A(t-s)} ds B q \right\rangle - \langle l^*, c_X^B \rangle \\ & \quad - \langle l^*, e^{At} M_{X_0}^A e^{At} l^* \rangle^{1/2} - \langle l^*, M_X^B l^* \rangle^{1/2} \\ & \quad - \|Q\|_2 \int_0^t \left\langle l^*, e^{A(t-s)} B B^T (e^{A(t-s)})^T l^* \right\rangle^{1/2} ds > 0. \end{aligned}$$

Simulation Experiment for Quadrotor

- Consider collision avoidance for quadrotors.

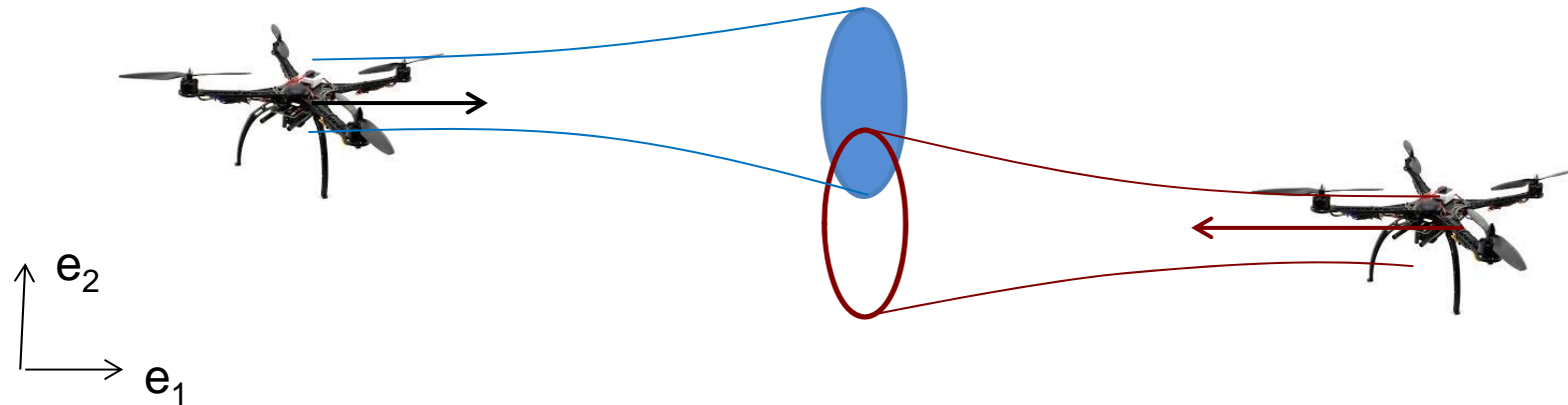
- The dynamics of the quadrotor⁶ is linearized around the hover mode.
- This reduces the dynamics to a ten dimensional $(x, y, z, \phi, \theta, u, v, w, p, q)$ system with three inputs (F, M_1, M_2) .



$$\begin{aligned} \dot{\xi} &= v \\ \dot{v} &= -g\mathbf{e}_3 + \frac{F}{m}R\mathbf{e}_3 \\ \dot{R} &= R\hat{\Omega} \\ \dot{\Omega} &= J^{-1}(-\Omega \times J\Omega + M) \end{aligned} \quad \Rightarrow \quad A = \begin{bmatrix} 0 & I & 0 & 0 \\ 0 & 0 & \begin{bmatrix} 0 & g \\ -g & 0 \\ 0 & 0 \end{bmatrix} & 0 \\ 0 & 0 & 0 & I \\ 0 & 0 & 0 & 0 \end{bmatrix}; \quad B = \begin{bmatrix} 0 & 0 \\ \begin{bmatrix} 0 \\ 0 \\ 1/m \end{bmatrix} & 0 \\ 0 & 0 \\ 0 & I_{2 \times 3} J^{-1} \end{bmatrix}$$

6. D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," *The International Journal of Robotics Research*, vol. 31, no. 5, pp. 664–674, 2012.

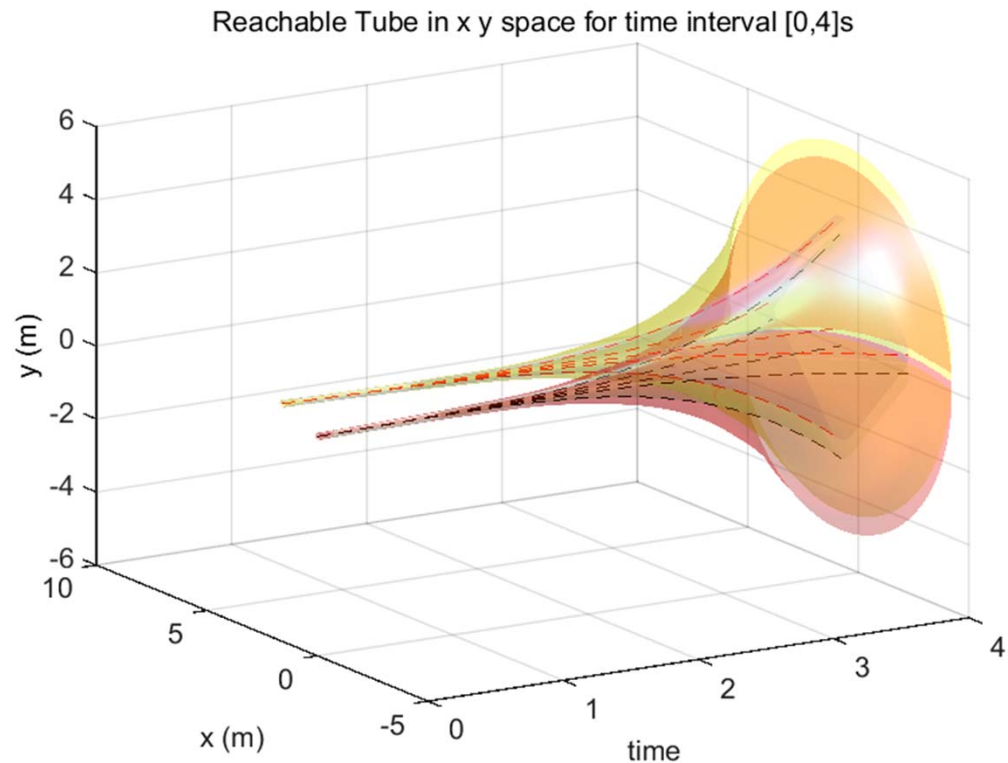
Simulation Setup for Quadrotor



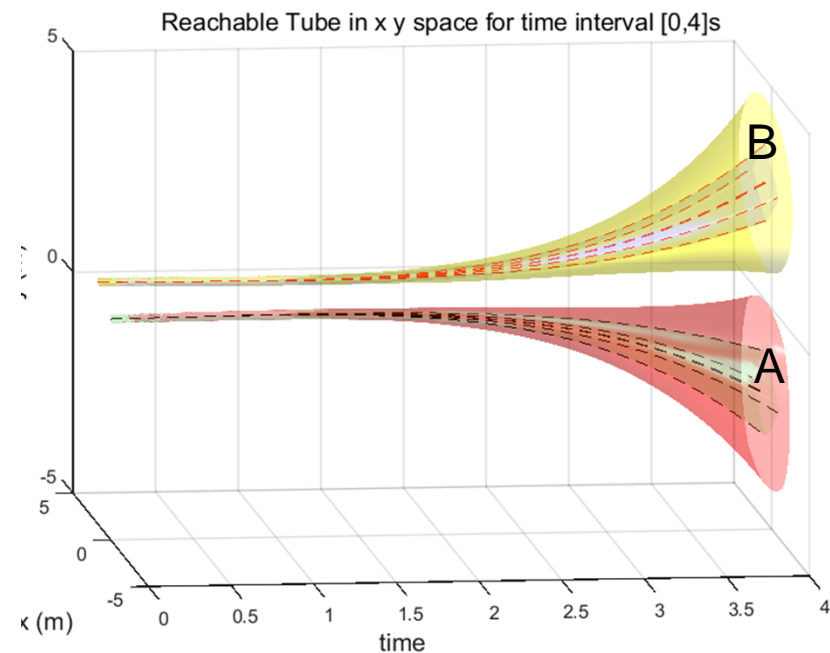
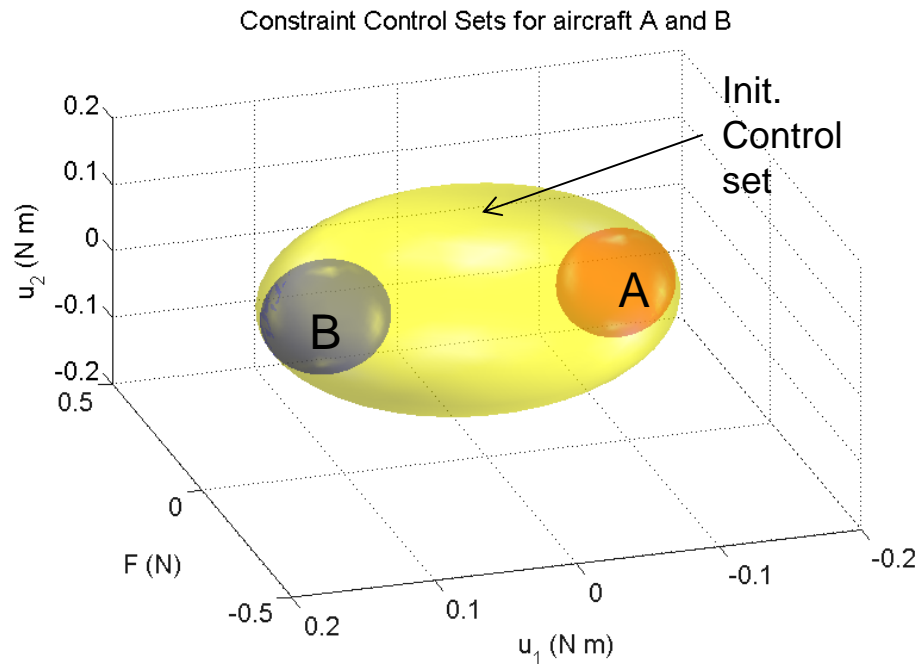
- Scenario is that two quadrotors are approaching each other from coordinates $(1.6, 0.5, 0)$ m and $(0, 0, 0)$ m with initial speed of $(-0.2, 0, 0)$ m/s, and $(0.2, 0, 0)$ m/s respectively.
- The separation requirement is 1 m. The time to collision is $\tau = 4$ s

Initial Reachable Set and Tube

- The reachable tubes of x , y position only in the bottom plot shows that the reachable sets overlap at the time of collision and earlier
- The darker colored ones in the center is the inner approximation of the reachable sets.
- The example trajectories are shown as dotted lines



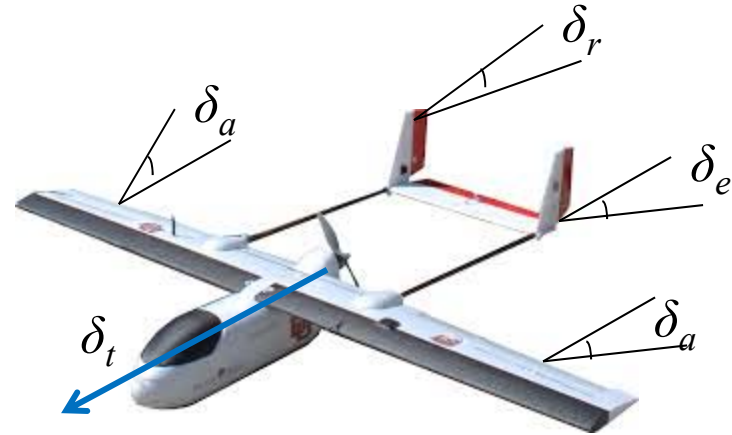
Collision Avoidance Control Set Synthesis



- On the left are the resulting **control** sets for aircraft A and B, on the right are the overall **reachable tubes** computed from the new control sets.
- Close examination of the reachable tubes also shows the resulting tubes are **collision free at every time instance**.
- Some of the example nonlinear trajectories are shown as dotted lines within the reachable tube.

Fixed-wing Aircraft

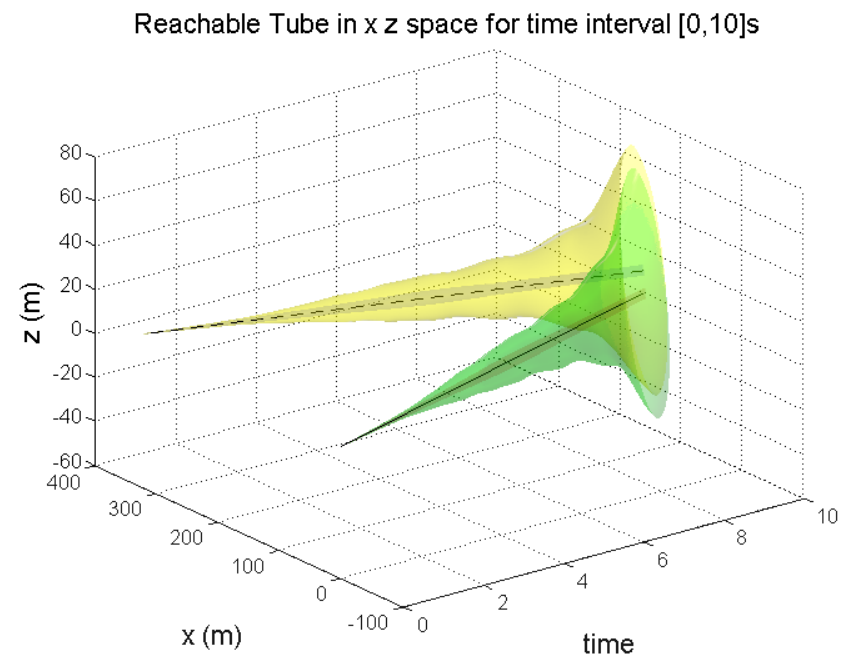
- More complex nonlinear dynamics due to aerodynamic force⁷
- Linearize around trimmed primitive motion (coordinated turn, straight leveled flight)
- Focus on **longitude motion** with **leveled cruise** mode.
- The states are (x, z, u, w, q, θ) , control inputs are (δ_p, δ_e)



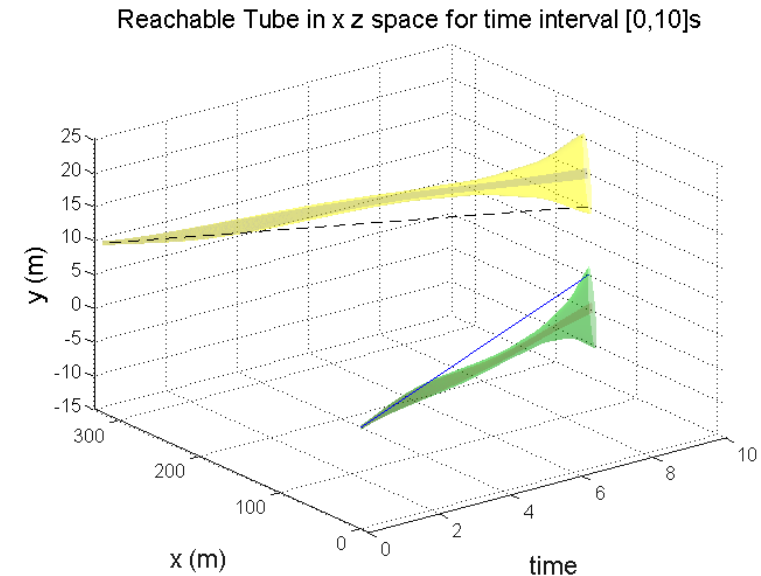
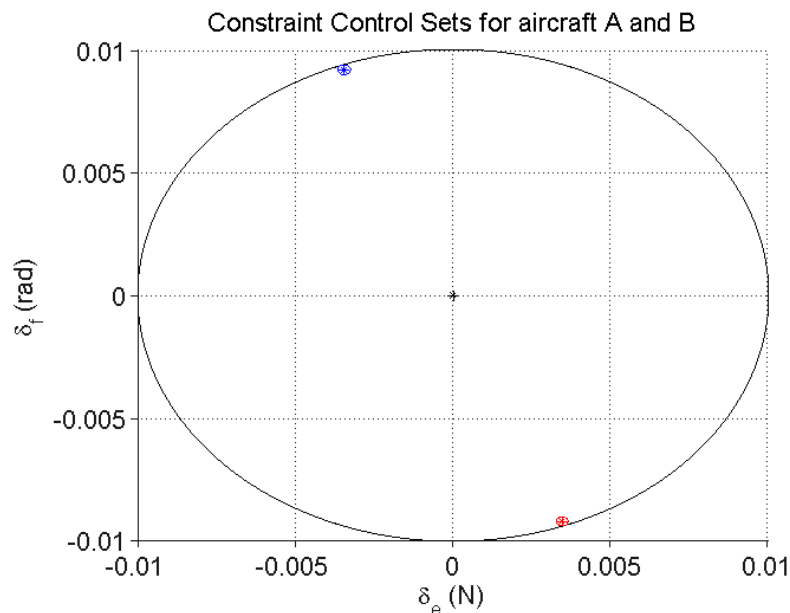
$$\begin{aligned} \dot{\xi} &= Rv_b \\ \dot{v}_b &= \begin{pmatrix} rv - qw \\ qw - ru \\ qu - pv \end{pmatrix} + \frac{1}{m} \begin{pmatrix} F_x \\ F_y \\ F_z \end{pmatrix} \Rightarrow A = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & u^* \\ 0 & 0 & X_u & X_w & X_q & -g \\ 0 & 0 & Z_u & Z_w & Z_q & 0 \\ 0 & 0 & M_u & M_w & M_q & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ X_{\delta_e} & X_{\delta_t} \\ Z_{\delta_e} & 0 \\ M_{\delta_e} & 0 \\ 0 & 0 \end{bmatrix} \\ \dot{R} &= R\hat{\Omega} \\ \dot{\Omega} &= J^{-1}(-\Omega \times J\Omega + T) \end{aligned}$$

Setup for Fixed-wing

- Scenario is that two aircraft are approaching each other from coordinates $(320, 10, 0)$ m and $(0, 0, 0)$ m with initial speed of $(-16, 0, 0)$ m/s, and $(16, 0, 0)$ m/s respectively.
- The separation requirement is 10 m. The time to collision is $\tau=10$ s.
- The nominal trajectories are the dotted lines in the centers of the reachable tubes



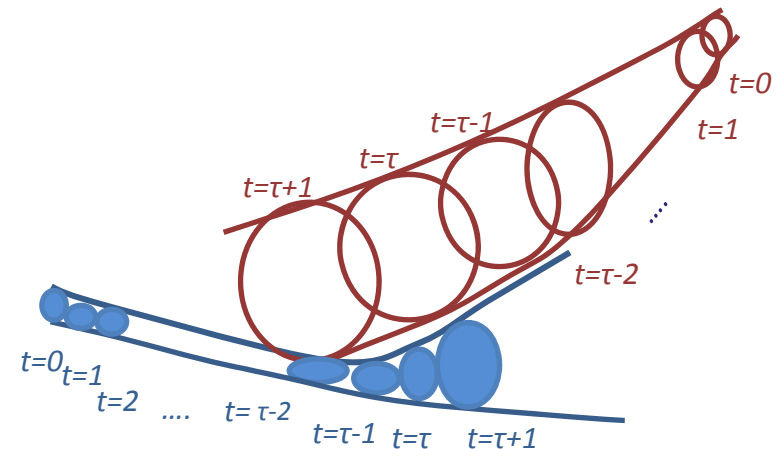
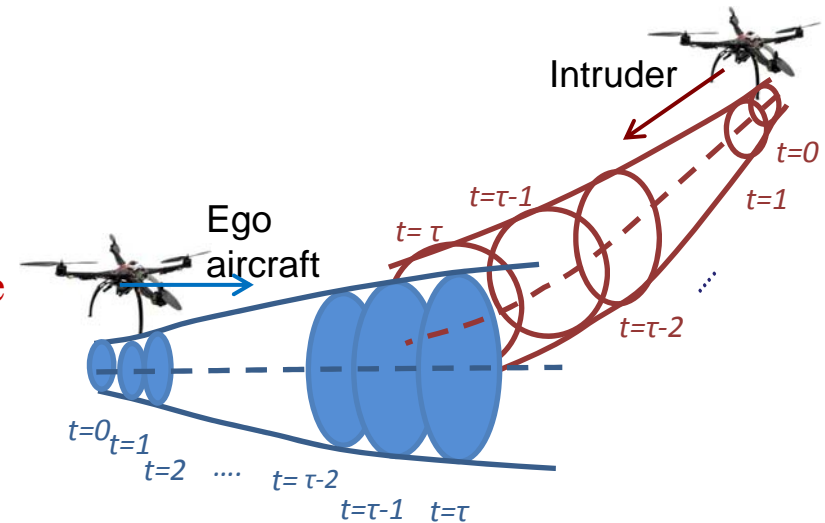
Result Control Set and Reachable Tubes



- The left figure shows the control sets of aircraft A and B, with the initial control set. The **constrained control sets** are **much smaller** compared to the original one, so only the centers of the ellipses are visible.
- The right figure shows the result reachable tube.

Collision Avoidance of Two UAVs with Time Varying Control Tube

- We seek a control set update rule design for ego aircraft in a non-collaborative setting
 - Guarantee collision avoidance **with reachable tube of the intruder aircraft**
 - The control constraint set should be time varying
 - Collision avoidance at every time instance
- Seek a tighter control constraint set such that
 - **Collision free** from predicted reachable set of intruder **at all times**
 - The control set should be as **large** as possible.
 - **Variation** in the control set should be small



Problem Setup

- Assume the control sets are scaled versions of the original set $\bar{\mathcal{U}}$

$$E(c_{\mathcal{U}}(t), \alpha(t)M_{\bar{\mathcal{U}}}) \subseteq \bar{\mathcal{U}} \quad \forall t$$

- The new optimization problem is the following

$$\begin{aligned} & \max_{q(t), \alpha(t)} \max_{l(t)} \int_{t=0}^T \mu(t) \alpha(t) dt \\ & \text{subject to} \quad \forall t \in [0, T] : \\ & \quad E(q(t), \alpha(t)M_{\bar{\mathcal{U}}}) \subseteq E(c_{\bar{\mathcal{U}}}, M_{\bar{\mathcal{U}}}) \\ & \quad -\rho(-l(t)|\mathcal{R}_x^i(t)) - \rho(l(t)|\mathcal{R}_x^e(t)) > 0 \end{aligned}$$

where $\mu(t) = T/(t+1)$.

- Let the control set update law be discrete, updated every δt .

$$\mathcal{U}(s) = E(q_d(k), \alpha_d(k)M_{\bar{\mathcal{U}}}) \quad \forall s \in [k\delta t, (k+1)\delta t)$$

- To simplify we assuming $l(t) = l^*(t)$ based on nominal trajectory, then the separation constraint becomes

$$\begin{aligned}
 & \langle l^*(t), c_X^i(t) \rangle - \langle l^*(t), M_X^i(t) l^* \rangle^{1/2} - \langle l^*(t), e^{At} c_{X_0}^e \rangle \\
 & \quad - \underbrace{\left\langle l^*(t), \int_0^t e^{A(t-s)} B q(s) ds \right\rangle}_{h(q,t)} - \langle l^*(t), e^{At} M_{X_0}^e e^{At} l^*(t) \rangle^{1/2} \\
 & \quad - \underbrace{\int_0^t \overbrace{\left\langle l^*(t), e^{A(t-s)} B M_{\bar{U}} B^T (e^{A(t-s)})^T l^*(t) \right\rangle^{1/2}}^{r(s,t)} \alpha^{1/2}(s) ds}_{g(a,t)} > 0.
 \end{aligned}$$

- Let the control set update law be discrete, updated every δt .

$$\mathcal{U}(s) = E(q_d(k), a^2(k) M_{\bar{U}}) \quad \forall s \in [k\delta t, (k+1)\delta t)$$

- At time $k\delta t$, the constraint on q and a , corresponding to h and g , are affine.

$$\begin{aligned}
 h(\mathbf{q}, k\delta t) &= \left\langle l^*(k\delta t), \sum_{i=1}^k \int_{(i-1)\delta t}^{i\delta t} e^{A(k\delta t-s)} B \mathbf{q}_i ds \right\rangle \\
 &= \left\langle l^*(k\delta t), \sum_{i=1}^k \sum_{j=1}^m \int_{(i-1)\delta t}^{i\delta t} e^{A(k\delta t-s)} b_j ds \mathbf{q}_{ji} \right\rangle \\
 &= \sum_{i=1}^k l^{*T}(k\delta t) P(i) \mathbf{q}_i \\
 g(\mathbf{a}, k\delta t) &= \sum_{i=1}^k \mathbf{a}_i \int_{(i-1)\delta t}^{i\delta t} r(s, k\delta t) ds
 \end{aligned}$$

Optimization Problem

- The overall Optimization can be rewritten as

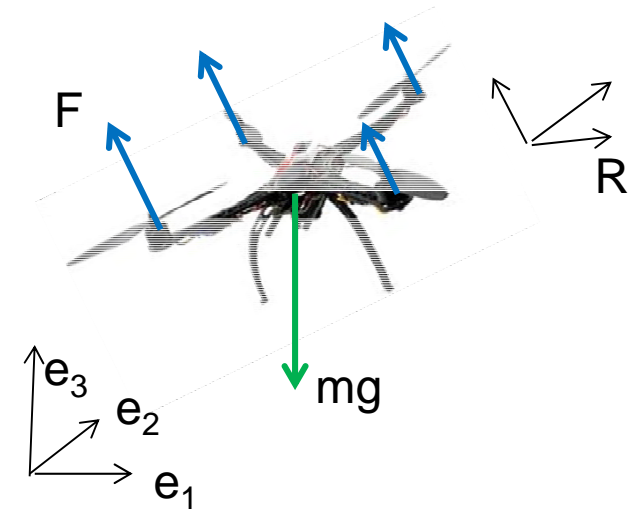
$$\begin{aligned}
 & \max_{\mathbf{a} \in \mathbb{R}^N, \mathbf{q} \in \mathbb{R}^{m \times N}} && \mu^T \mathbf{a} - \|H^T \mathbf{q}\|_F \\
 & \text{subject to} && \forall k \in 1, 2 \dots N : \\
 & && \lambda(k) > 0 \\
 & && \begin{bmatrix} 1 - \lambda(k) & 0 & (\mathbf{q}_k - c_{\bar{U}})^T \\ 0 & \lambda(k)I & \mathbf{a}_k(M_{\bar{U}})^{1/2} \\ \mathbf{q}_k - c_{\bar{U}} & \mathbf{a}_k(M_{\bar{U}})^{1/2} & M_{\bar{U}} \end{bmatrix} \succeq 0 \\
 & && \rho(-l(k\delta t)|\mathcal{R}_x^i(k\delta t)) - \rho(l(k\delta t)|\mathcal{R}_x^e(k\delta t)) > 0
 \end{aligned}$$

- \mathbf{a} is the scaling of the control sets over time. \mathbf{q} captures the center.
- $H^T \mathbf{q}$ captures the variations in discretized control set center.
- $\|X\|_F$ represents the Frobenius norm, i.e. $\|X\|_F = \left(\sum_j^m \sum_i^n |x_{ij}|^2 \right)^{1/2}$

Simulation Experiment for Quadrotor

- Consider collision avoidance for quadrotors.

- The dynamics of the quadrotor⁶ is linearized around the hover mode.
- This reduces the dynamics to a ten dimensional $(x, y, z, \phi, \theta, u, v, w, p, q)$ system with three inputs (F, M_1, M_2) .



$$\begin{aligned} \dot{\xi} &= v \\ \dot{v} &= -g\mathbf{e}_3 + \frac{F}{m}R\mathbf{e}_3 \\ \dot{R} &= R\hat{\Omega} \\ \dot{\Omega} &= J^{-1}(-\Omega \times J\Omega + M) \end{aligned} \quad \longrightarrow \quad A = \begin{bmatrix} 0 & I & 0 & 0 \\ 0 & 0 & \begin{bmatrix} 0 & g \\ -g & 0 \\ 0 & 0 \end{bmatrix} & 0 \\ 0 & 0 & 0 & I \\ 0 & 0 & 0 & 0 \end{bmatrix}; \quad B = \begin{bmatrix} 0 & 0 \\ \begin{bmatrix} 0 \\ 0 \\ 1/m \end{bmatrix} & 0 \\ 0 & 0 \\ 0 & I_{2 \times 3} J^{-1} \end{bmatrix}$$

6. D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," *The International Journal of Robotics Research*, vol. 31, no. 5, pp. 664–674, 2012.

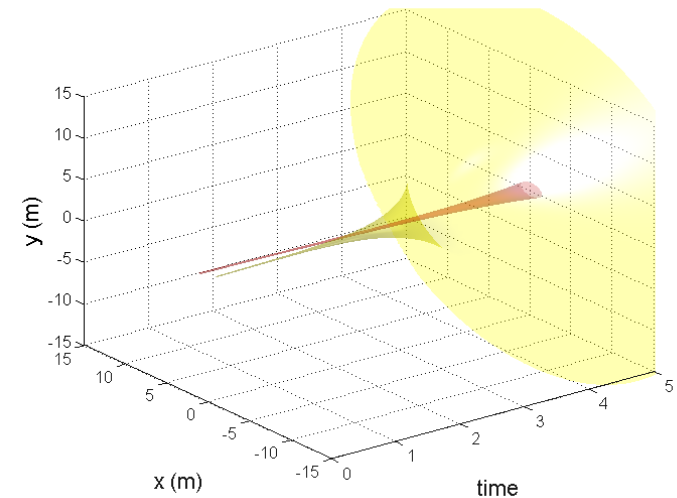
Simulation Setup

- We will consider collision avoidance for a **quadrotor**.
 - The dynamics of the quadrotor is **linearized around the hover**.
 - This reduces the dynamics to a ten dimensional system with three inputs.
- Scenario is that two quadrotors are approaching each other from coordinates $(1, 0.5, 0)$ m and $(-1, 0, 0)$ m with initial speed of $(-0.2, 0, 0)$ m/s, and $(0.2, 0, 0)$ m/s respectively.
- The separation requirement is 1 m. It is fairly easy to estimate the collision time which is $\tau = 5$ s

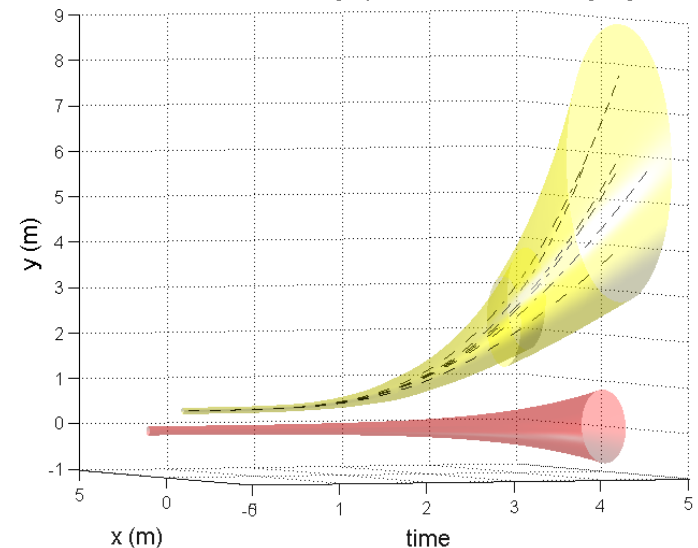
Simulation for Quadrotor

- Similar setup as the quadrotor one earlier, both UAV heading towards each other
- Top plot shows the initial reachable tubes. Red tube is for the intruder vehicle, while the yellow one is for the ego vehicle.
- Bottom plot shows the resulting reachable tube. The exemplary trajectories of full nonlinear dynamics are included as dashed lines.

Reachable Tube in x y space for time interval [0,5]s



Reachable Tube in x y space for time interval [0,5]s



■ Summary

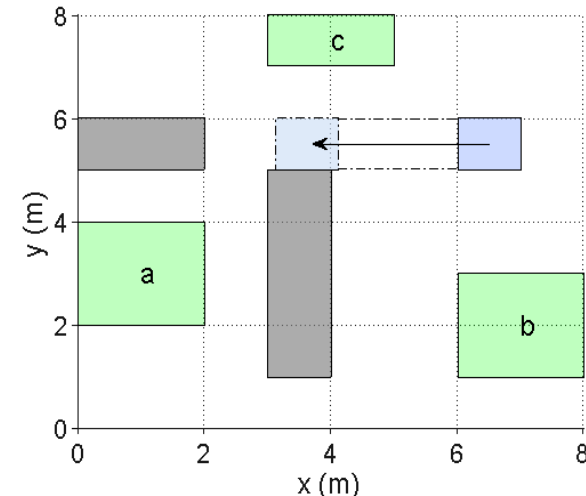
- Developed several set-valued control designs for collision avoidance problems.
- Provable safety guarantee by construction.
- Optimization can be done online and in real-time.

■ Future work

- Collision avoidance between multiple agents
- Physical UAV implementation

Part II: Motion Planning for Temporal Logics with Finite Time Constraints

- **Problem:** How to generate trajectory/path based on **temporal specifications** such as ordering, repetition, safety?
- **State of the art:** motion planning with temporal constraints without duration, such as Linear Temporal Logic (LTL).
- Two methods for **timed temporal logics**, such as Metric Temporal Logic(MTL):
 - An optimization based method⁸
 - A timed-automata based method⁹



Task: Always visiting area a,b,c and stay there for at least 2s. Always avoiding obstacles

8. Y. Zhou, D. Maity and J. S. Baras, "Optimal Mission Planner with Timed Temporal Logic Constraints", Proceedings of 2015 European Control Conference, Linz, Austria, pp. 759-764, July 15-17, 2015.
9. Y. Zhou, D. Maity, and J. S Baras. "Timed automata approach for motion planning using metric interval temporal logic.", accepted to 2016 European Control Conference, Aalborg Denmark, June 29 - July 1, 2016.

Robotic Motion Planning Problem

Given:

A dynamic workspace (environment),

A **time constrained task** (ϕ),

A cost function.

Objective:

Find the suitable control input such that the robot completes the **given task** and **minimizes** the cost function.

Constraints:

Avoiding collisions with all **static and moving obstacles** in the workspace.

Linear Temporal Logic



- Effective in representing **high-level complex logical tasks**: has been used extensively in robotic motion planning
- Commonly specifies safety and surveillance related mission
- Composed of the following,
 - Atomic proposition: π
 - Logic operators:
 - \neg (negation), \vee (disjunction), \wedge (conjunction) and \Rightarrow (implication)
 - Temporal operators:
 - \bigcirc (next), \mathcal{U} (until), \diamond (eventually) and \square (always)

Slide 87

D2

May delete this slide

Dipankar, 6/9/2016

LTL Examples

- **Condition:** “After sensing M2, go to room A3 and stay there, otherwise stay at start (A0)”

$$(A0 \mathcal{U} \text{sensor}(M2) \wedge \text{sensor}(M2) \Rightarrow \Diamond \Box A3)$$

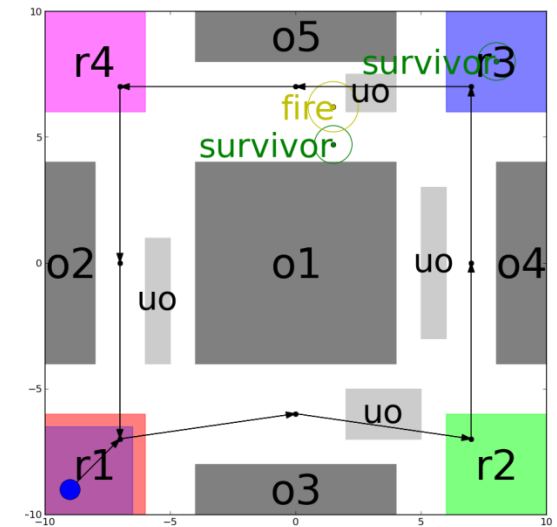
- **Surveillance:** “Always gather data at gathering locations and then upload the data, repeat infinitely many times”

$$(\Box \Diamond (\text{gather} \Rightarrow \Diamond \text{upload}))$$

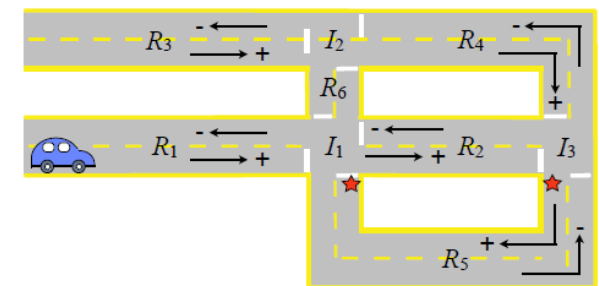
- **Safety:** “Always avoid obstacle region A1 or A2, while achieve task ϕ ”

$$\Box (\neg (A1 \vee A2) \wedge \phi)$$

- **Traffic Rule**



Reactive sampling based
temporal logic planning¹⁰



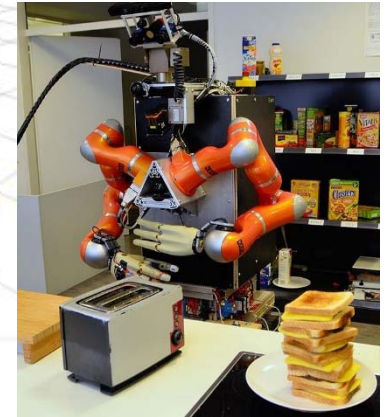
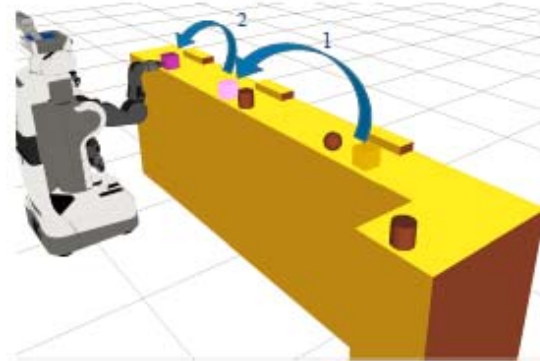
Receding horizon temporal
logic planning¹¹

10. Vasile, Cristian Ioan, and Calin Belta. "Reactive sampling-based temporal logic path planning." Robotics and Automation (ICRA), 2014 IEEE Intern. Conference on. IEEE, 2014.

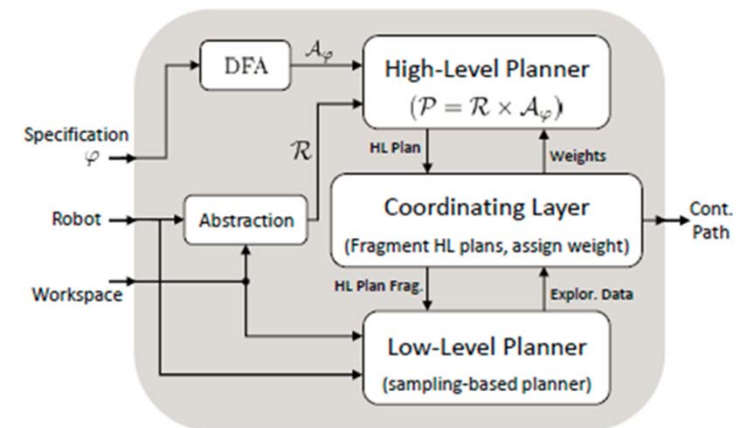
11. Wongpiromsarn, Tichakorn, Ufuk Topcu, and Richard M. Murray. "Receding horizon temporal logic planning." *Automatic Control, IEEE Transactions on* 57.11 (2012): 2817-2830.

A Robotic Motion Planning Example

- Manipulation task planning²
 - First, take food to customers and bring the empty plates back to the preparation area. Next, show the tip jar to the ones whom have already finished eating.
- The question is how fast to take the food to the customers, or what is a good time to ask for the tips from the customers. So timing aspects are important.
- Many robotic tasks require finite time constraints.
- **LTL is unable to address finite time constraints and hence we need MITL.**



Towards manipulation planning with temporal logic specifications²



2. K. He, M. Lahijanian, L. E. Kavraki, and M. Y. Vardi, "Towards manipulation planning with temporal logic specifications," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, 2015,

Metric Temporal Logic (MTL) and Time Constrained Task

Definition: The syntax of **MTL**¹² (**MITL**¹³) formulas are defined according to the following grammar rules:

$$\phi ::= \top \mid \pi \mid \neg\phi \mid \phi \vee \phi \mid \phi U_I \phi$$

where $I \subseteq [0, \infty]$ is **an interval** with end points in $\mathbb{N} \cup \{\infty\}$ and the end points have to be distinct. $\pi \in \Pi$ is the atomic proposition.

More sophisticated MTL (MITL) operators can be **derived** using the grammar defined above; such as: **always** in $I_1 \equiv \top U_{I_1} \top$, **eventually always** $\Diamond_{I_1} \Box_{I_2}$ etc.

12. R. Koymans, "Specifying real-time properties with metric temporal logic," *Real-time systems*, vol. 2, no. 4, pp. 255–299, 1990.

13. R. Alur, T. Feder, and T. A. Henzinger, "The benefits of relaxing punctuality," *J. of the ACM (JACM)*, vol. 43, no. 1, pp. 116–146, 1996.

Part II.1: Mixed Integer Optimization Based Method

$$\begin{aligned} \min_u \quad & J(x(t, u), u(t)) \\ \text{Subject to} \quad & x(t + 1) = f(t, x(t), u(t)) \\ & \mathbf{x}_{t_0} \models \varphi \end{aligned}$$

Remarks:

The task φ may be a **finite duration** task within an **infinite** time horizon task such as surveillance, periodic tasks etc.

From MTL Constraints to Linear Constraints

A polygon can be represented as intersections of several half-planes.

The constraint $z_i^t = 1$ iff $h_i^T x(t) \leq k_i$ is enforced by the linear constraints:

$$\begin{aligned} h_i^T x(t) &\leq k_i + M(1 - z_i^t) \\ h_i^T x(t) &\geq k_i - Mz_i^t + \epsilon \end{aligned} \tag{1}$$

where M is a very large positive number and ϵ is a very small positive number, and $z_i^t \in \{0,1\}$.

Let $\mathcal{P} = \bigcap_{i=1}^n H_i$ be a polygon with $H_i = \{x \mid h_i^T x \leq k_i\}$.

Define $P_t^{\mathcal{P}} = \bigwedge_{i=1}^n z_i^t$, then $P_t^{\mathcal{P}} = 1$ iff $x(t) \in \mathcal{P}$.

MTL Rules as Linear Constraints

$x_t \models \mathcal{P}$ iff $x(t) \in \mathcal{P}$ is equivalent to $P_t^{\mathcal{P}} = 1$.

The linear-integer constraints equivalent to $P_t^{\mathcal{P}} = 1$ are given in (1)

Negation Operator:

$$\varphi = \neg p: \quad P_t^{\varphi} = 1 - P_t^p$$

MTL Rules as Linear Constraints (Cont.)

Conjunction Operator:

$$\varphi = \bigwedge_{i=1}^m p_i : \quad P_t^\varphi \leq P_t^{p_i}, \quad i = 1, \dots, m$$
$$P_t^\varphi \geq 1 - m + \sum_{i=1}^m P_t^{p_i}.$$

Disjunction Operator:

$$\varphi = \bigvee_{i=1}^m p_i : \quad P_t^\varphi \geq P_t^{p_i}, \quad i = 1, \dots, m$$
$$P_t^\varphi \leq \sum_{i=1}^m P_t^{p_i}.$$

MTL Rules as Linear Constraints (Cont.)

Let $t \in \{0, 1, \dots, N - t_2\}$, N is the total time horizon for planning.

Eventually within $[t_1, t_2]$: $\varphi = \Diamond_{[t_1, t_2]} p$:

$$P_t^\varphi \geq P_\tau^p, \quad \tau \in \{t + t_1, \dots, t + t_2\}$$

$$P_t^\varphi \leq \sum_{\{\tau=t+t_1\}}^{\{t+t_2\}} P_\tau^p.$$

Always within $[t_1, t_2]$: $\varphi = \Box_{[t_1, t_2]} p$

$$P_t^\varphi \leq P_\tau^p, \quad \tau \in \{t + t_1, \dots, t + t_2\}$$

$$P_t^\varphi \geq \sum_{\{\tau=t+t_1\}}^{\{t+t_2\}} P_\tau^p + t_1 - t_2.$$

MTL Rules as Linear Constraints (Cont.)

Until within $[t_1, t_2]$: $\varphi = pU_{[t_1, t_2]}q$

$$j \in \{t + t_1, \dots, t + t_2\}$$

$$a_{tj} \leq P_j^q,$$

$$a_{tj} \leq P_k^p, \quad k \in \{t, \dots, j - 1\},$$

$$a_{tj} \geq P_j^q + \sum_{k=t}^{j-1} P_k^p + t - j,$$

$$P_t^\varphi \leq \sum_{j=t+t_1}^{t+t_2} a_{tj},$$

$$P_t^\varphi \geq a_{tj},$$

Modification of Original Problem into MILP

$$\begin{aligned} \min_{u, z_0, \dots, z_N \in \{0,1\}^p} \quad & J(x(t, u), u(t)) \\ \text{Subject to} \quad & x(t+1) = f(t, x(t), u(t)) \\ & L(x(t), z_t, t) \leq 0 \quad \forall t \in [0, N] \end{aligned}$$

The timed temporal constraint $\mathbf{x}_{t_0} \models \boldsymbol{\varphi}$ can be converted into the linear and integer constraints.

Remark:

If $J(\cdot, \cdot)$ $f(\cdot, \cdot, \cdot)$ are linear functions of $x(t)$ and $u(t)$, then entire problem will be a **Mixed-Integer Linear Optimization Problem**.

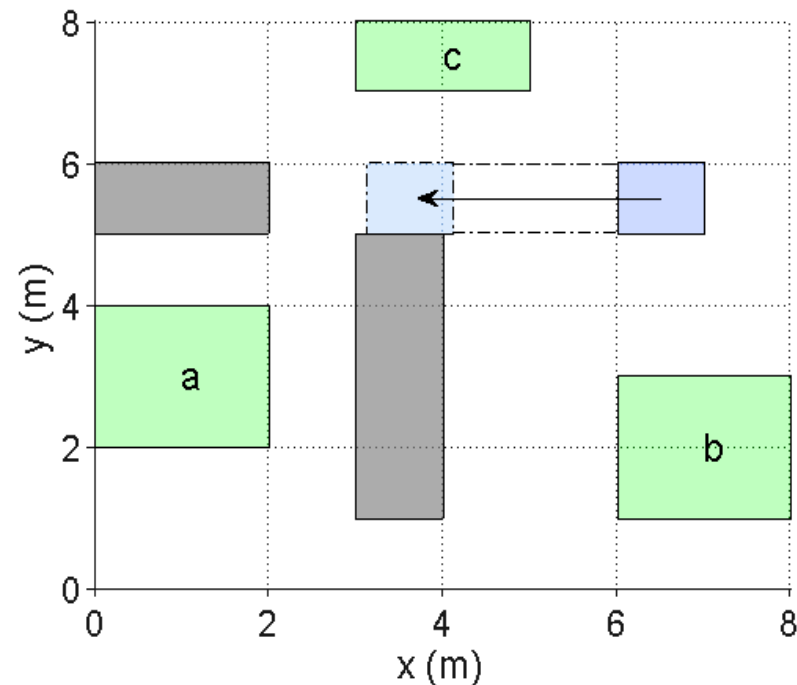
Experimental Setup

$$J(x, u) = \sum_{t=0}^N |u(t)|$$

- Quadrotor:
 - Linearization is the same as the reachable set computation. The system is linearized about hover.
- Car:
 - Linearized about different discrete angles $\hat{\theta}$

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos(\theta) & 0 \\ \sin(\theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \Rightarrow A = \begin{bmatrix} 0 & 0 & -\hat{u}_1 \sin(\hat{\theta}) \\ 0 & 0 & \hat{u}_1 \cos(\hat{\theta}) \\ 0 & 0 & 0 \end{bmatrix}; B = \begin{bmatrix} \cos(\hat{\theta}) & 0 \\ \sin(\hat{\theta}) & 0 \\ 0 & 1 \end{bmatrix}$$

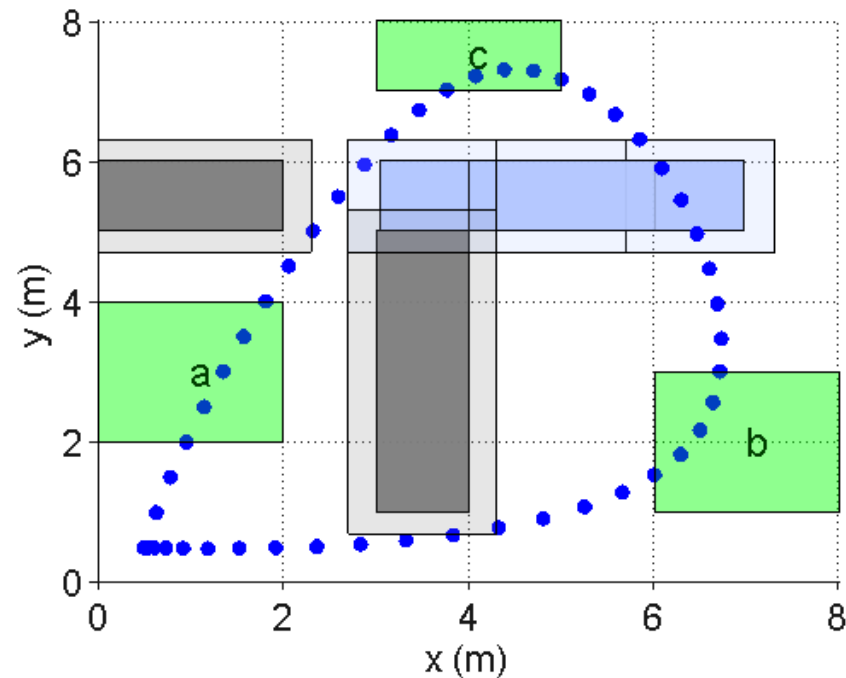
- **Task 1:**
 - The specification requires the autonomous vehicle to eventually visit area 'a', 'b', and 'c', and stay there for at least 2 time units, while avoiding obstacles.



The grey boxes represent fixed obstacles, while the blue one is a moving obstacle with fixed speed.

- Specification in MTL

$$\phi_1 = \Diamond \Box_{[0,2]} A \wedge \Diamond \Box_{[0,2]} B \\ \wedge \Diamond \Box_{[0,2]} C \wedge \Box \neg O$$
- The resulting trajectory for the linearized quadrotor dynamics, projected in 2D.

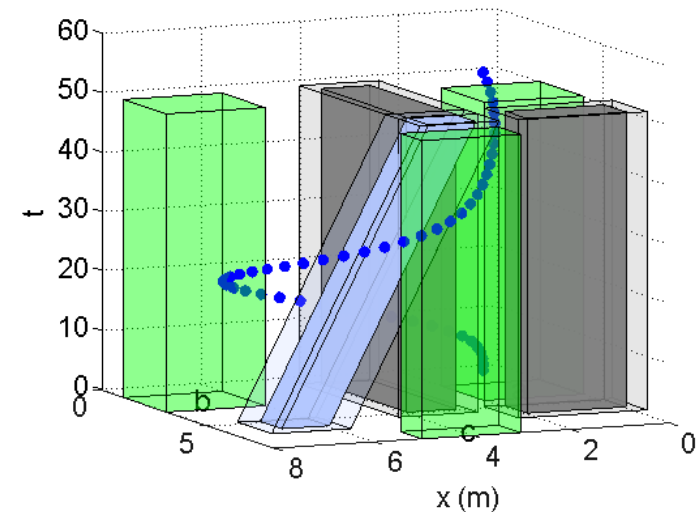
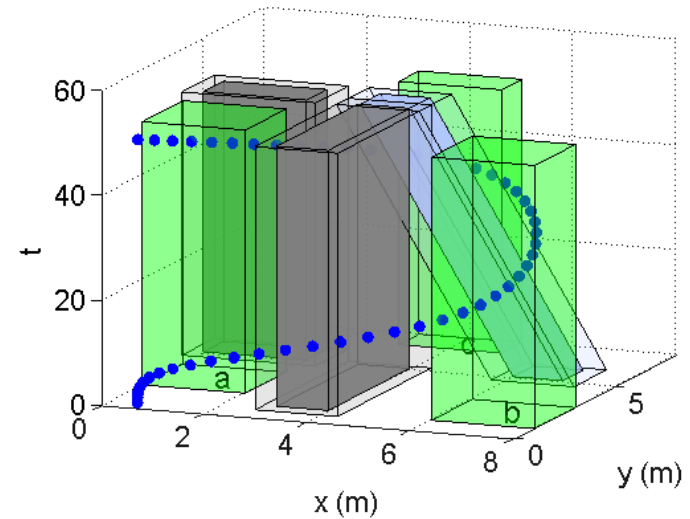


2D projection of the trajectory of the quadrotor satisfying the task.

- Specification in MTL

$$\phi_1 = \Diamond \Box_{[0,2]} A \wedge \Diamond \Box_{[0,2]} B \\ \wedge \Diamond \Box_{[0,2]} C \wedge \Box \neg O$$

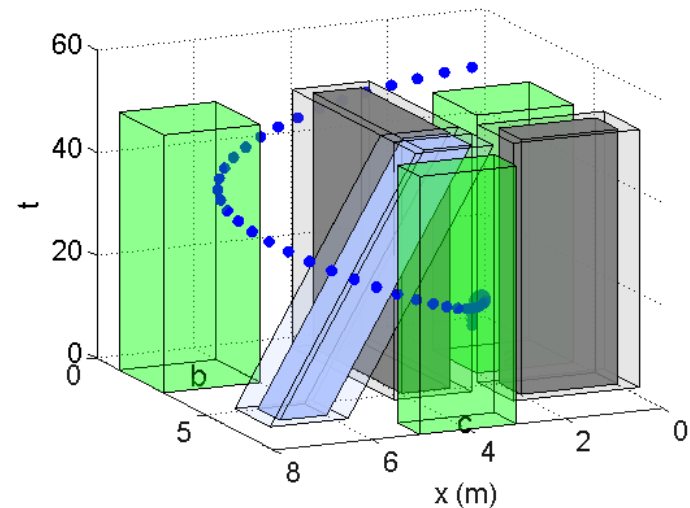
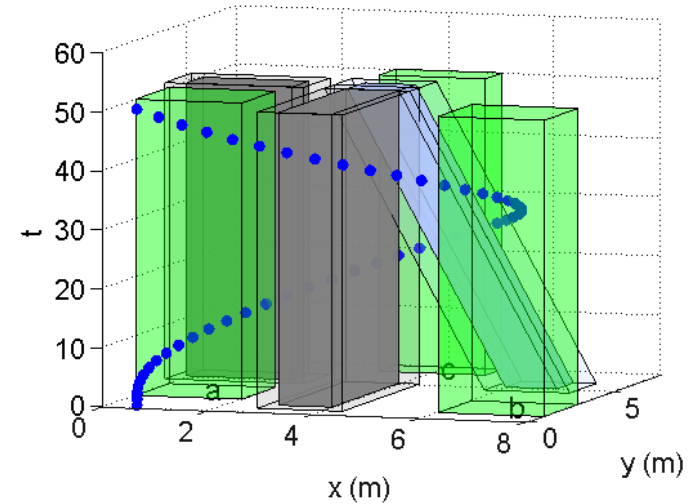
- 3D Trajectory
 - The trajectory avoids the obstacle region in time and space



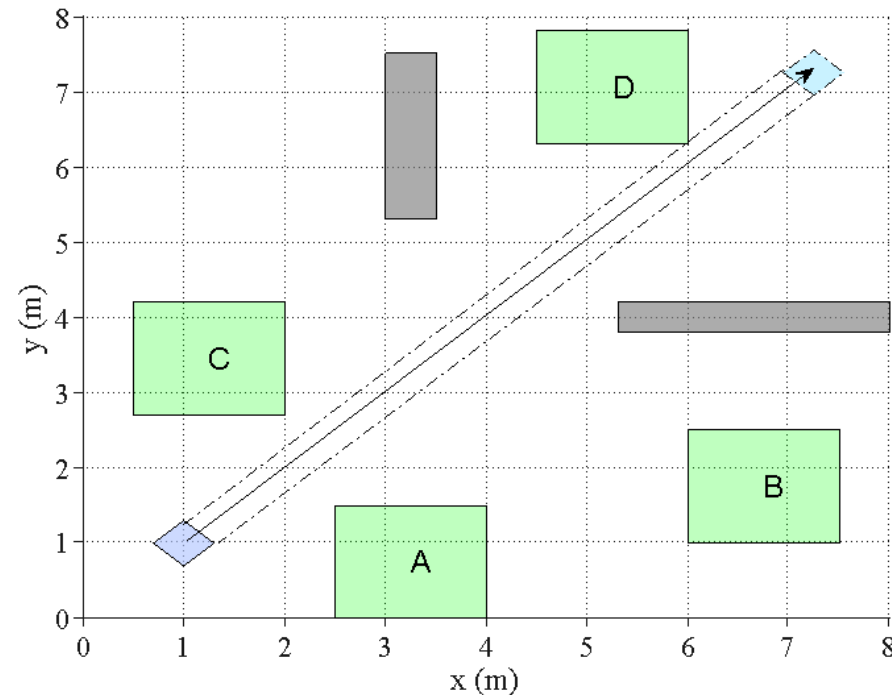
- If ordering between area A and area B is restricted, for example, area A has to be visited first, the modified specification in MITL will be the following

$$\phi_2 = \Diamond \Box_{[0,2]} A \wedge \Diamond \Box_{[0,2]} B \wedge \Diamond \Box_{[0,2]} C \\ \wedge \Box \neg O \wedge \neg B U_{[0,N]} A$$

- 3D Trajectory from two different points of view is shown in right.



- **Task 2:**
 - The specification requires the autonomous vehicle to eventually visit area A, B, C and D, and stay there for at least 2 time units, while avoiding obstacles.

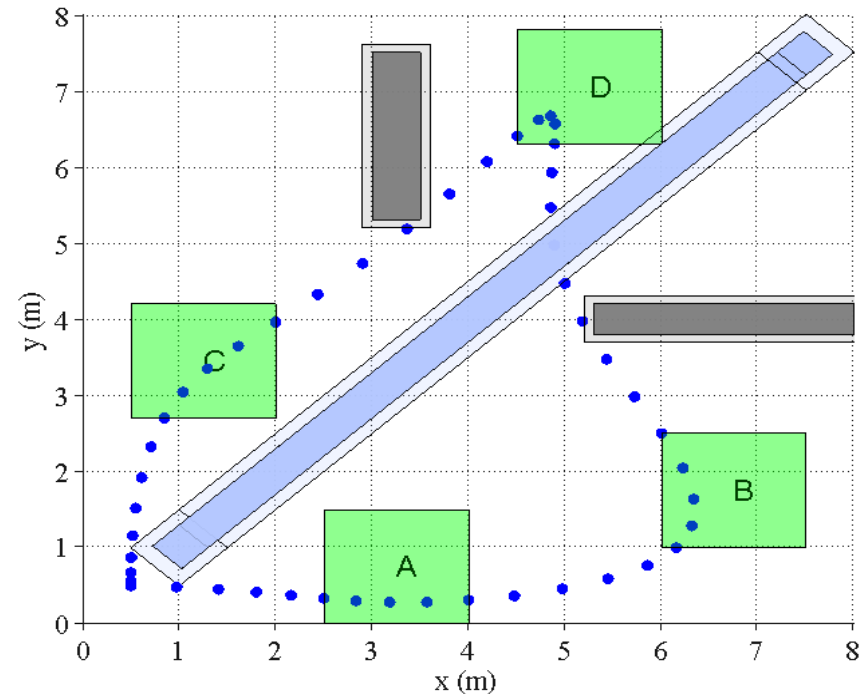


A much crowded workspace. The grey boxes represent fixed obstacles, while the blue one is a moving obstacle with fixed speed.

- Specification in MTL

$$\phi_3 = \Diamond \Box_{[0,2]} A \wedge \Diamond \Box_{[0,2]} B \\ \wedge \Diamond \Box_{[0,2]} C \wedge \Diamond \Box_{[0,2]} D \wedge \Box \neg O$$

- The result for linearized quadrotor dynamic projected in 2D is shown in right as blue dots



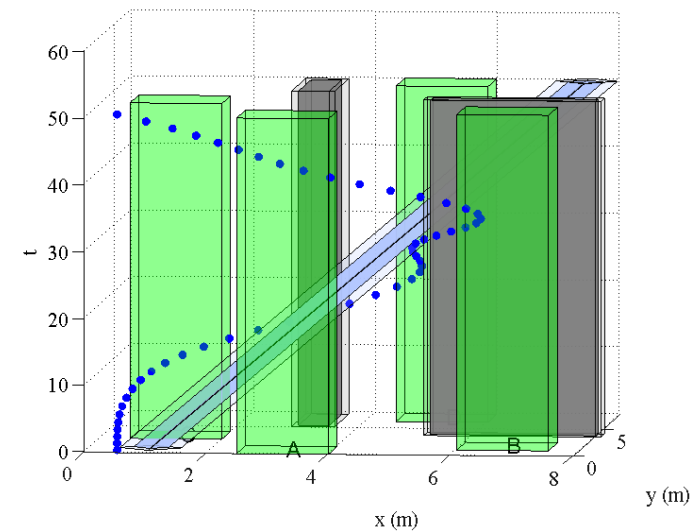
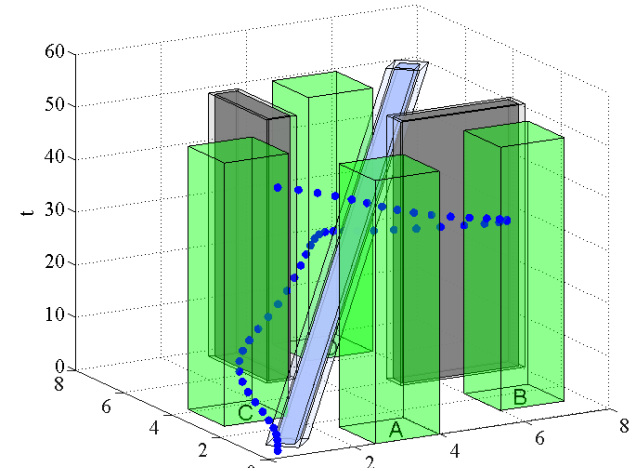
2D projection of the trajectory of the quadrotor satisfying the task.

- Specification in MTL

$$\begin{aligned}\phi_3 = & \Diamond \Box_{[0,2]} A \wedge \Diamond \Box_{[0,2]} B \\ & \wedge \Diamond \Box_{[0,2]} C \wedge \Diamond \Box_{[0,2]} D \wedge \Box \neg O\end{aligned}$$

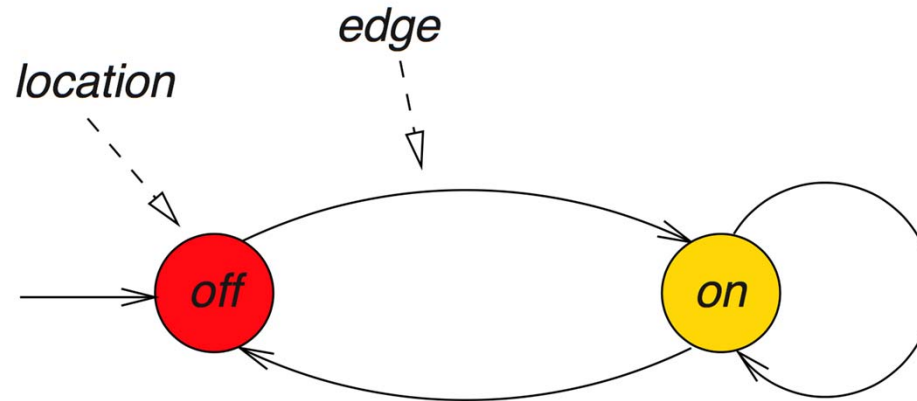
- 3D Trajectory

- The trajectory avoids the obstacle region in time and space.



Part II.2: Timed Automata Based Method

Automata Generated from LTL



- A program graph with **locations** and **edges**
- A location is labeled with the **valid atomic propositions**
- Taking an edge is **instantaneous**, i.e, consumes no time

Remark: An LTL formula can equivalently be represented by a Büchi Automaton

Robotic Motion Planning Problem

Given:

A dynamic workspace (environment),

A **time constrained task** (ϕ),

A cost function.

Objective:

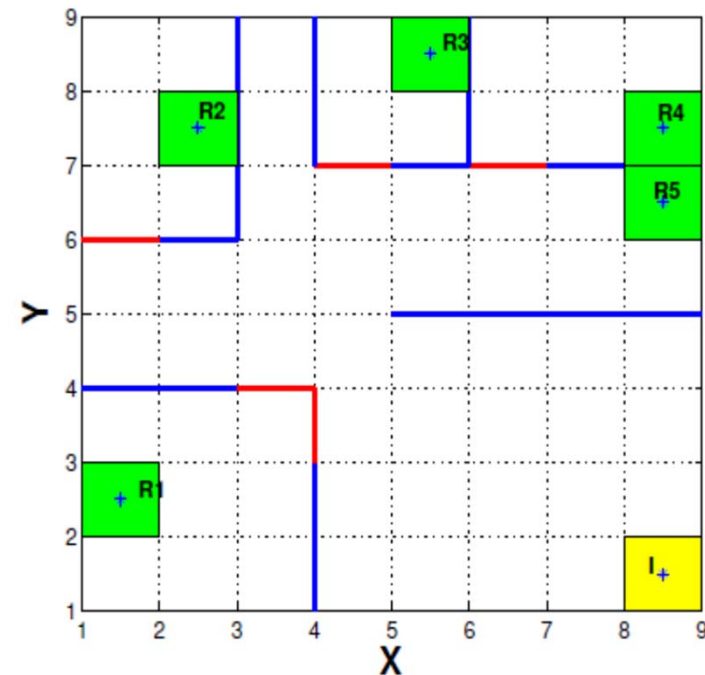
Find the suitable control input such that the robot completes the **given task** and **minimizes** the cost function.

Constraints:

Avoiding collisions with all **static and moving obstacles** in the workspace.

Robot Motion Planning Problem

Example: Starting from I , visit R_3 within the time interval T_1 , visit R_4 within time interval T_2 ; before visiting R_3 or R_4 , robot must visit R_2 . Eventually visit R_1 and R_5 , and complete the whole task in the least time.

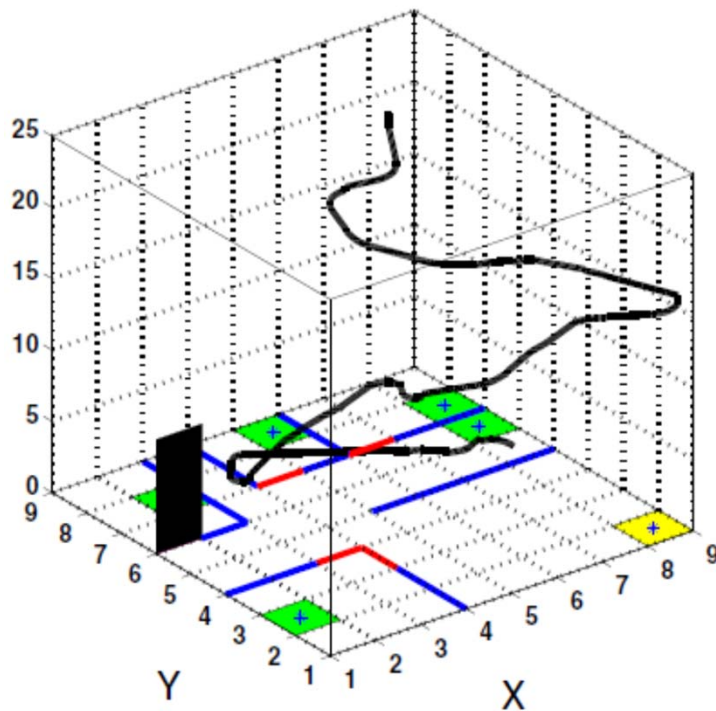


The workspace of the robot.

Discretized Workspace and Time

Dynamical Properties:

1. The doors can be open or closed at different times.
2. Presence of moving obstacles.



The discretized workspace-time.

Labeling the Time Varying Environment

Labeling function:

$$F: S \times T \rightarrow 2^{\Pi}$$

S: Workspace of the Robot.

T: Time interval.

$\Pi = \{\pi_i \mid i = 1, 2, \dots, n\}$: Set of propositions; e.g. free space, regions (R_i), obstacles etc.

Labeling the Time Varying Environment

Time independent labeling function:

$$F_I: S \rightarrow 2^{\Pi_T}$$

$\Pi_T = \{\pi_{k,I} \mid \pi_k \in \Pi, I \in T\}$: Time dependent set of propositions

$\pi_{k,t} \in F_I(s)$ if and only if $\pi_k \in F(s, t)$

Temporal Logic Based Motion Planning

Review: Linear Temporal Logic (LTL)

The syntax of LTL formulas are defined according to the following grammar rules:

$$\varphi ::= \top \mid \pi \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi U \varphi \mid \varphi R \varphi$$

Boolean Operators: AND (\wedge), OR (\vee), True (\top), False (\perp), negation (\neg)

Temporal Operators: Eventually (\diamond), Until (U), Always (\square) etc.

Temporal Logic Based Motion Planning

Extended Linear Temporal Logic (LTL)

The extension of the LTL grammar is given by

$$\varphi ::= \varphi U_I \varphi \mid \varphi R_I \varphi$$

U_I means: ‘Until’ operation has to be true within time interval I .

Other operators: Eventually within I ($\Diamond_I \equiv \top R_I$)

Always in I ($\Box_I \equiv \perp U_I$)

Our LTL formula:

$$(\Diamond_{T_1} R_3) \wedge (\Diamond_{T_2} R_4) \wedge (\neg(R_3 \vee R_4) U R_2) \wedge (\Diamond R_1) \wedge (\Diamond R_5) \wedge (\Box(\neg O))$$

From Extended LTL Operator to Classical LTL Operator

Proposition1: The LTL formula $\varphi_1 U_I \varphi_2$ is equivalent to $\varphi_{1,[0,b]} U \varphi_{2,I}$, where $I = [a, b]$.

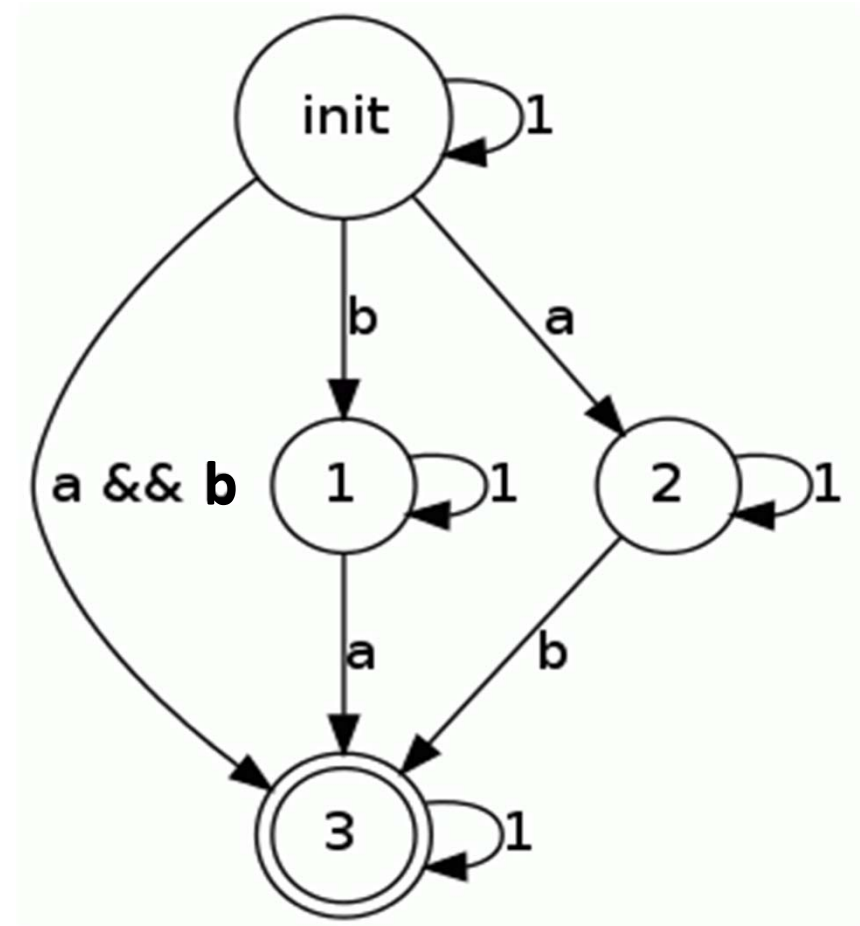
Proposition2: The LTL formula $\varphi_1 R_I \varphi_2$ is equivalent to $\varphi_{1,I} R \varphi_{2,[0,\infty)}$.

LTL Formula to Büchi Automata

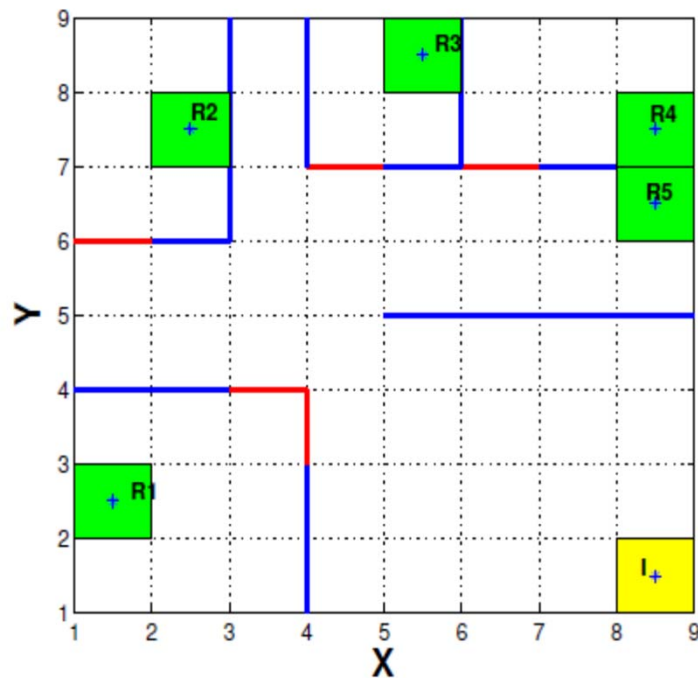
Example:

Formula: $(\Diamond a) \wedge (\Diamond b)$

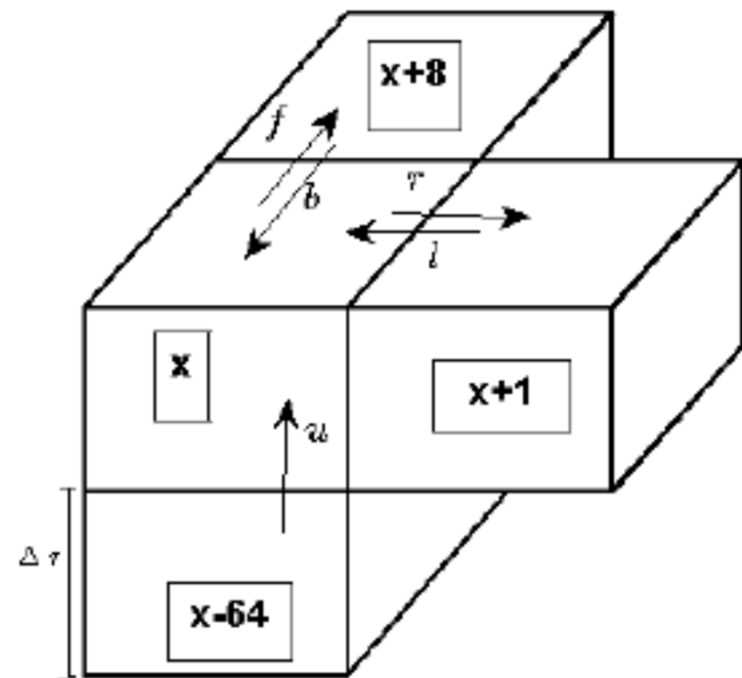
Formula: aUb



Workspace as State Transition System



The workspace of the robot.

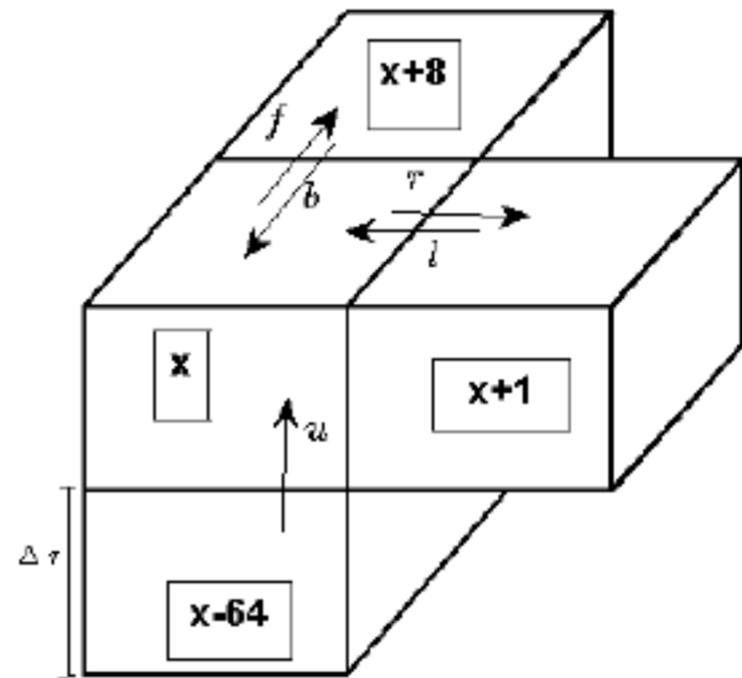


Transitional relationship among the blocks in discretized workspace-time.

Workspace as State Transition System

Definition: The equivalent FTS(ε) of the discretized workspace-time is given by a tuple
 $\{Q, Q_0, \Pi_I, \mathcal{A}, TS, \rightarrow_\varepsilon, Q_F\}$

Remark: $p: \mathbb{N} \rightarrow Q$ is a path with corresponding action sequence $a_0 a_1 \dots a_n$, then
 $p(0) \in Q_0, (p(i+1), p(i)) \in \rightarrow_\varepsilon$
 $p(i+1) = TS(p(i), a_i)$



Transitional relationship among the blocks in discretized workspace-time.

Product Automata and the Equivalent Graph

The product automata $\mathcal{P} = \{\mathcal{S}_P, \mathcal{S}_{0P}, 2^{\Pi_I}, \delta_P, \mathcal{P}_F\}$,
where $\mathcal{S}_P = \mathcal{S}_\varphi \times \mathcal{Q}$

$$\mathcal{S}_{0P} = \mathcal{S}_{0\varphi} \times \mathcal{Q}_0$$

$$\delta_P: \mathcal{S}_P \times 2^{\Pi_I} \rightarrow 2^{\mathcal{S}_P}$$

$$\mathcal{P}_F = \mathcal{S}_{F\varphi} \times \mathcal{Q}_F$$

- By construction the language of this product automata is $L(\mathcal{P}) = L(\varphi) \cap L(\mathcal{E})$.

Planning Problem as a Graph Search Problem

- Let the graph equivalent to the product automata be $\mathcal{G}(V, E, W)$
- $V \equiv \mathcal{S}_P$ is the set of nodes.
- $E \equiv \delta_P$ is the set of edges.
- W is the weight associated with edges
- In this case time spent can be put as a weight on the edges.
- The resulting problem is a shortest path graph search problem.

Introduction of the Robot Dynamics

For our simulation purposes we consider a non-holonomic dynamics of a robot:

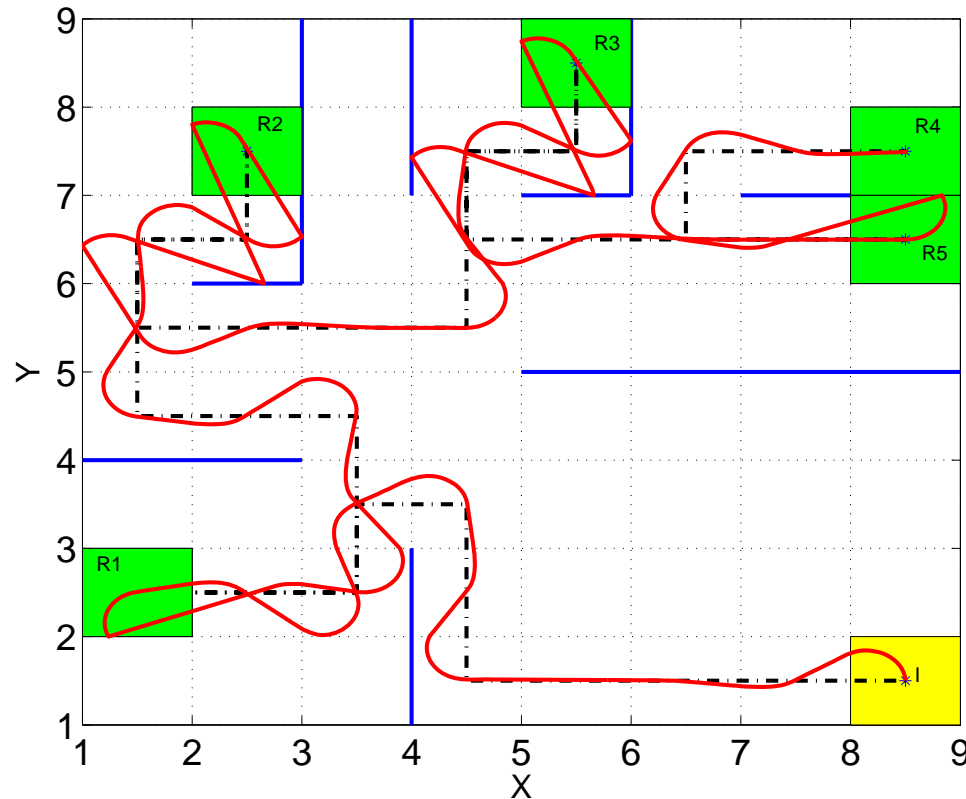
$$\begin{bmatrix} \dot{x} \\ y \\ \theta \end{bmatrix} = \begin{bmatrix} v \cos \theta \\ v \sin \theta \\ \omega \end{bmatrix}.$$

Inputs: v and ω .

Finding Control Inputs

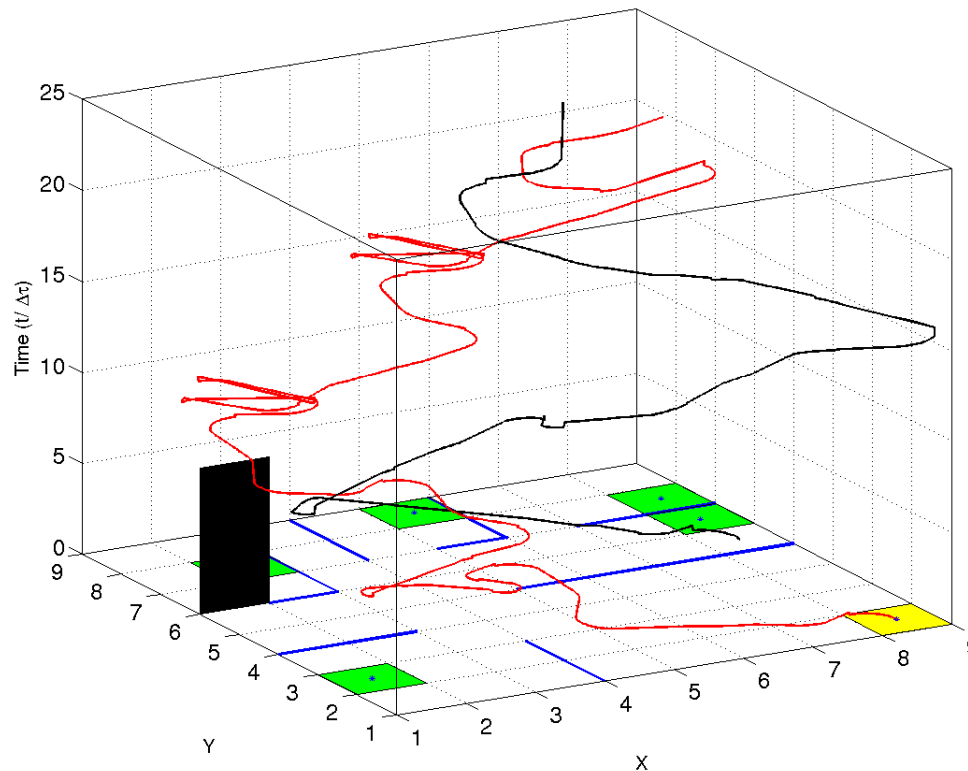
- Find control inputs to maneuver the robot along the discrete path obtained by solving the graph search problem.
- Constraints:
While moving from one cell to another, the robot must be confined within these two cells only.

Simulation Result



Projected continuous and discrete trajectories. (Dashed black line is the discrete path and the red curve is the corresponding continuous trajectory.)

Simulation Result



Continuous trajectory in spacetime. (Red curve is the trajectory generated for the robot and the black curve is the obstacle trajectory.)

Summary

- Framework for time constrained planning problems.
- Extension of the LTL operators provides machinery to express a time constrained LTL based task.
- Converting an extended LTL operator to a regular LTL operator is also provided so that available model-checking tools e.g. SPIN, NuSMV can be directly used to convert the LTL formula to a Büchi automaton.

- Incorporate uncertainties into the system.
- Online planning and re-planning (incremental planning) procedures.
- Multi robot systems and collaborative framework.

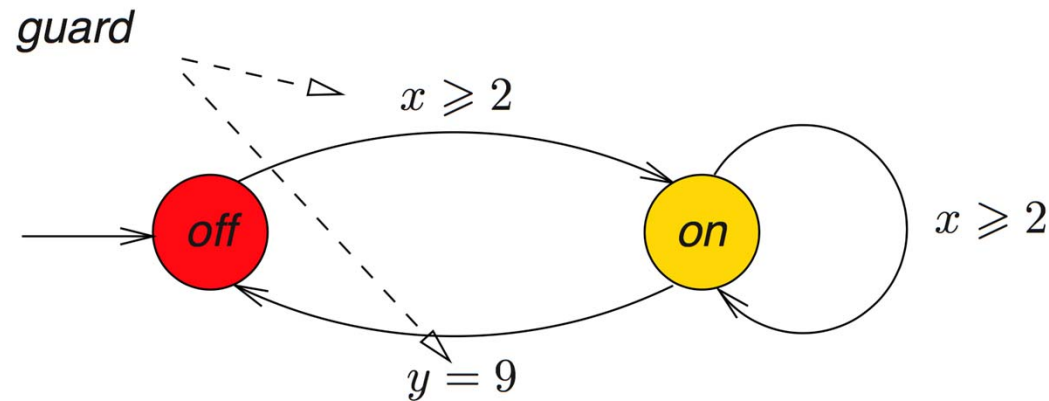
Metric Temporal Logic

Metric Temporal Logic (MTL) deals with model checking under timing constraints.

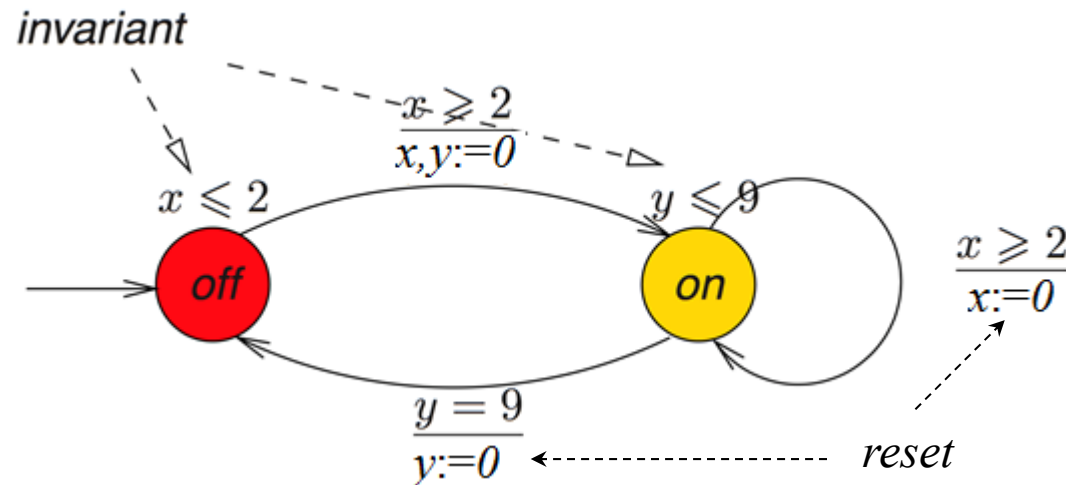
Complexity of MTL model checking is undecided and MITL (Metric Interval Temporal Logic), a subset of MTL, has the complexity of EXPSpace-complete whereas LTL is PSPACE-complete.

Signal Temporal Logic (STL), similar to MTL also performs model checking with timing constraints. Based on the fact that LTL is computationally less expensive and that there is good availability of tools to check LTL specifications, the previous method translates a bounded time high level specification to a purely LTL specification.

Adding Clock Variables and Guard conditions: Timed Automata



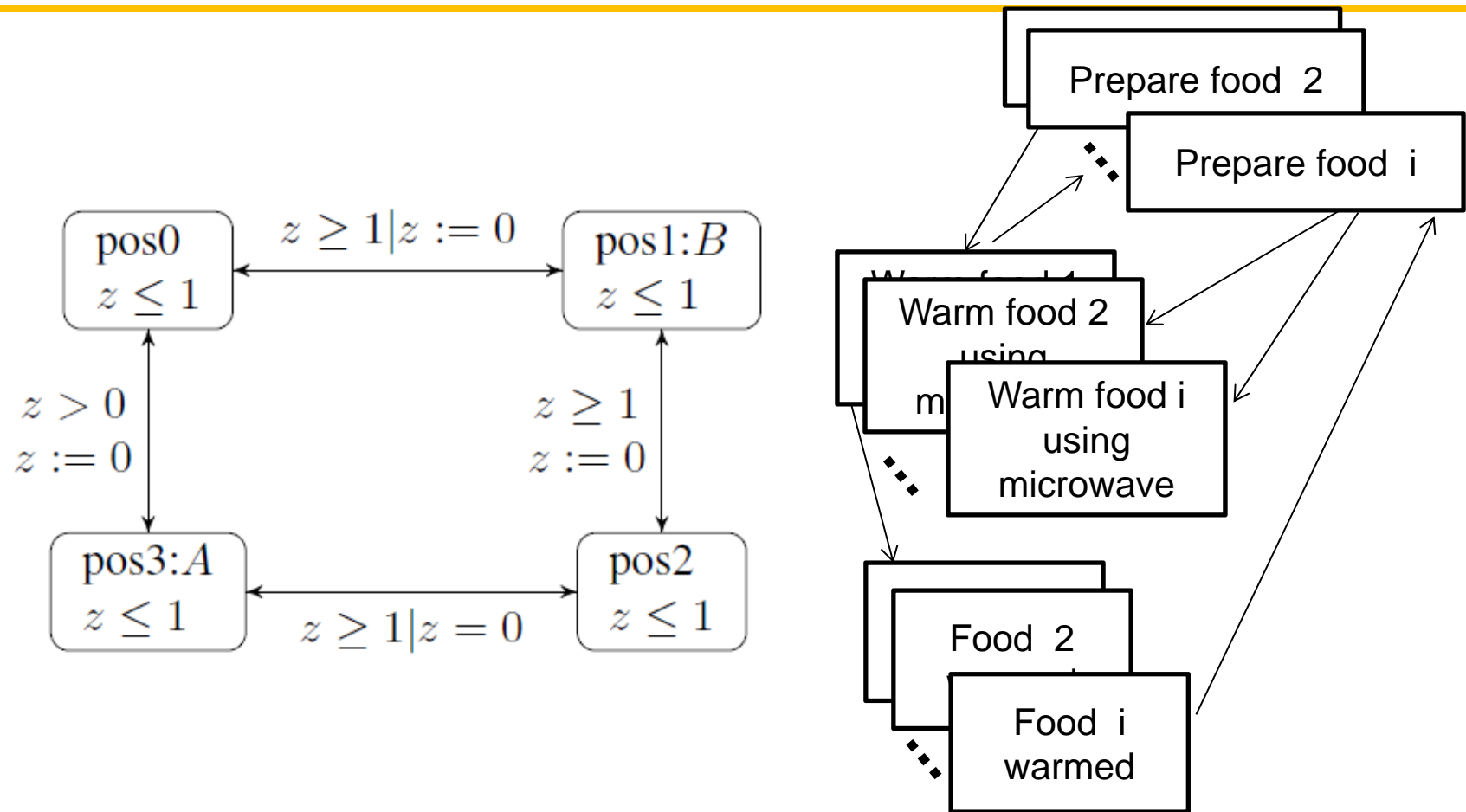
- Equipped with real-valued **clocks** x, y, z, \dots
- Clocks advance implicitly, all at the **same speed**
- Traversing an edge is **instantaneous**, i.e., consumes no time
- Guards indicate when an edge **may** be taken



- Clocks can be **reset** when taking an edge
- Assumption: all clocks are **zero** when entering the initial location initially
- A location invariant specifies the **amount of time that may be spent in a location**

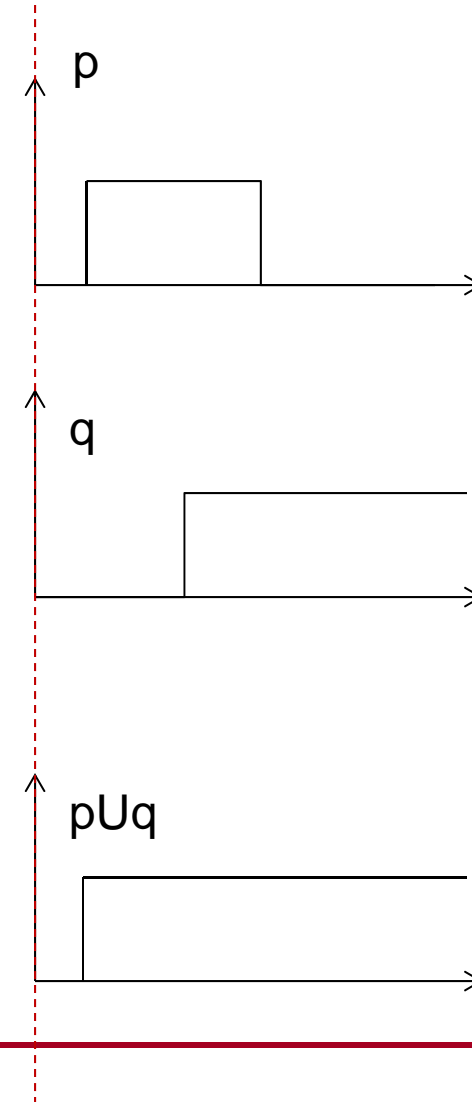
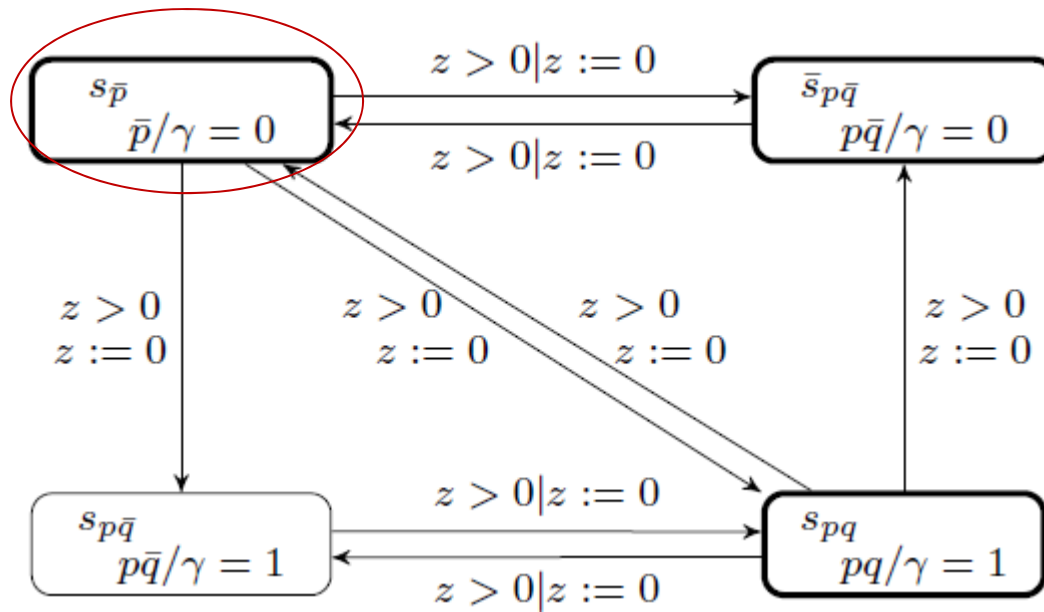
Remark: An MITL formula can equivalently be represented by a Timed Automaton

Timed Automata and Robotic Action

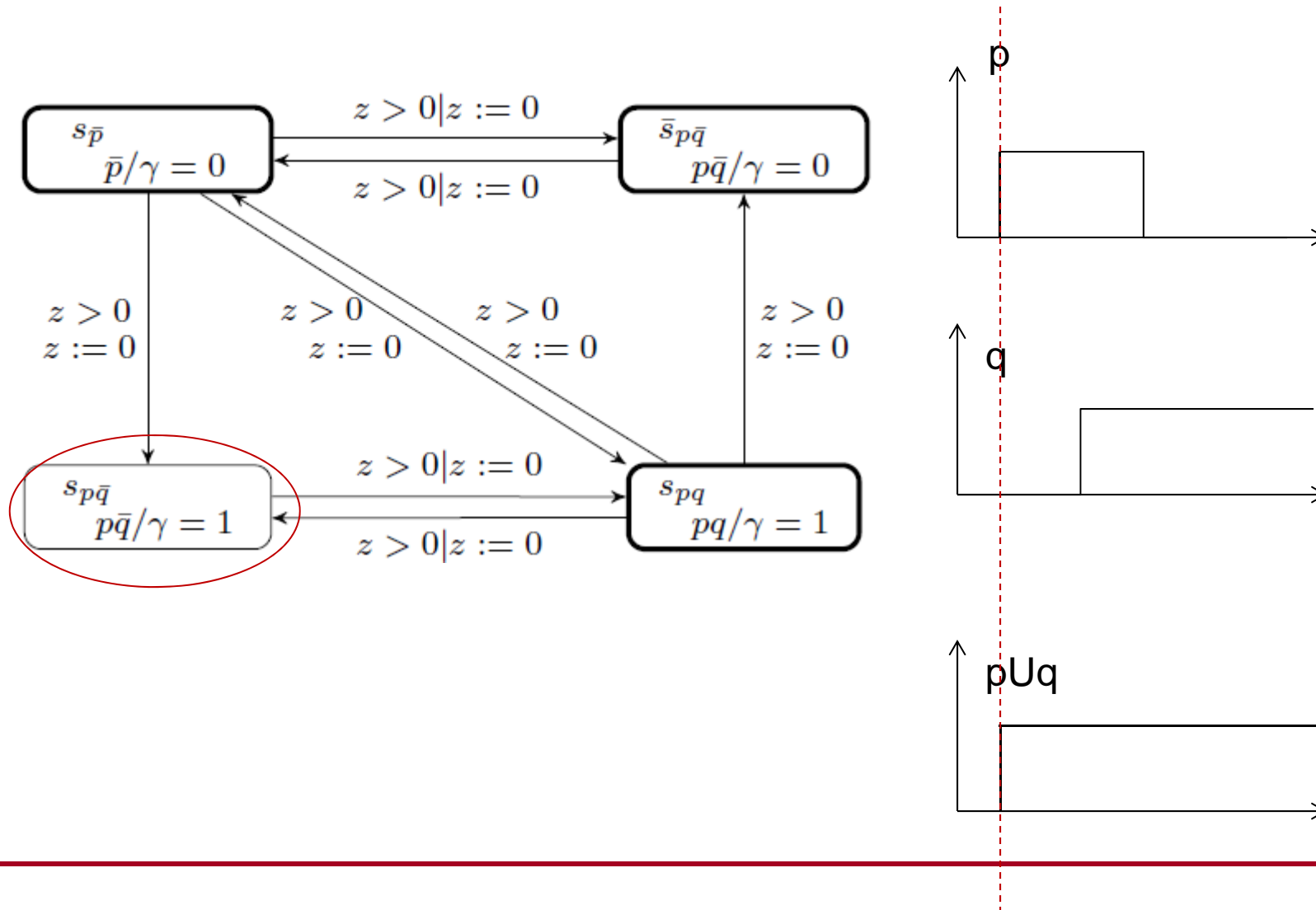


Robotic motion including manipulation task can
be captured in timed automata

Automaton Representation of Until Operator



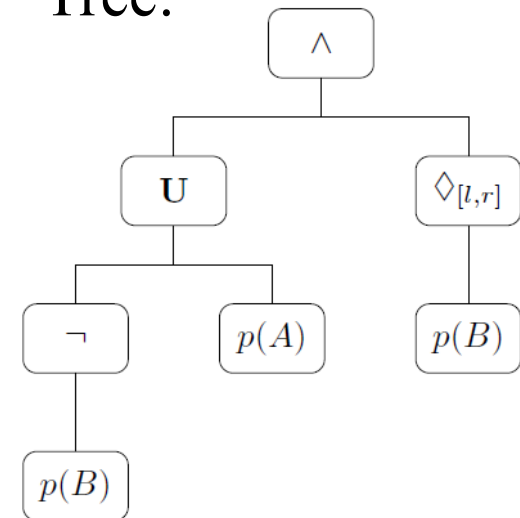
Automaton Representation of Until Operator



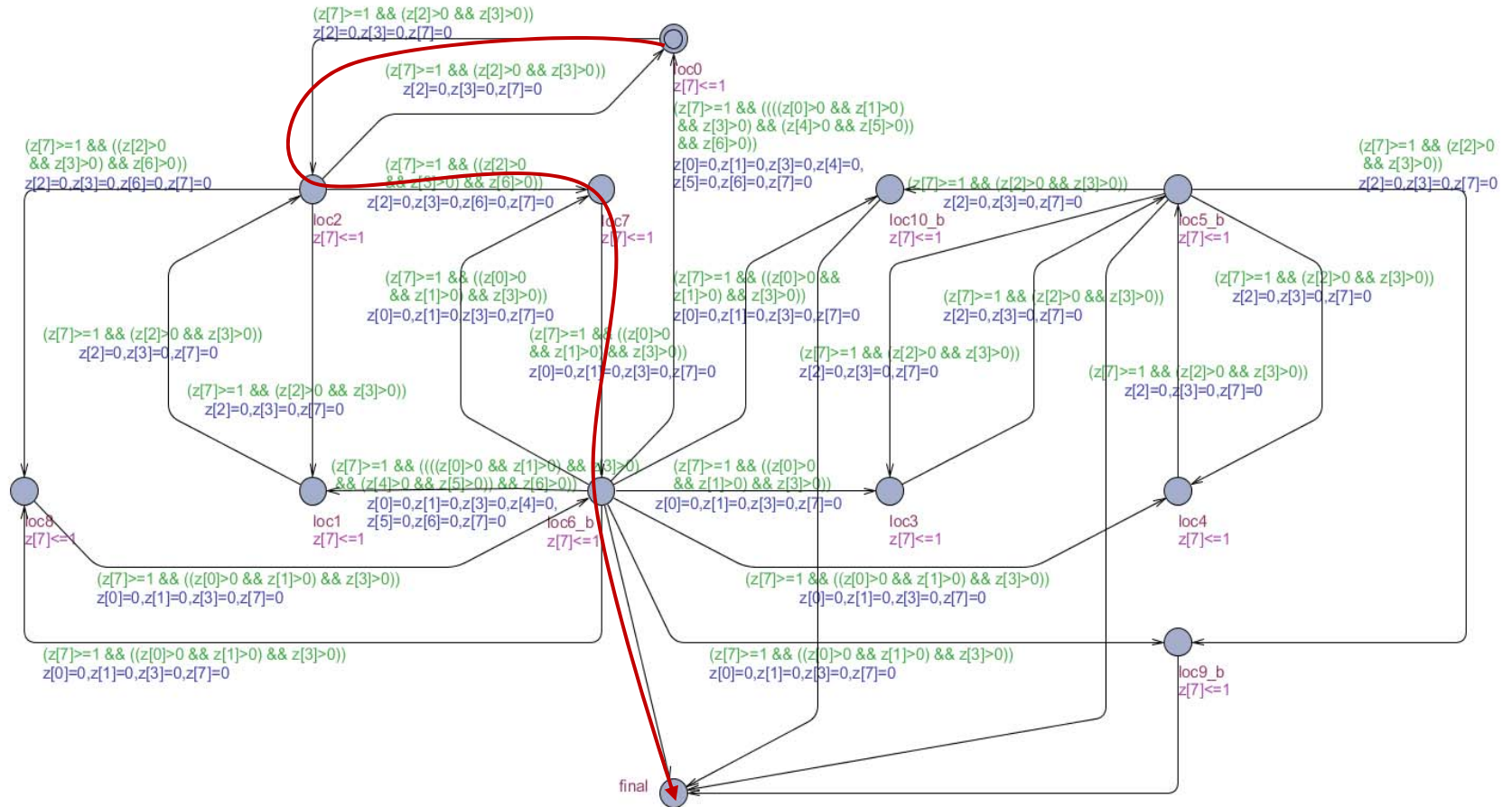
Timed Automata Based Planning Example

- Convert temporal logic formula to a timed automaton
 - Represent temporal logics as a tree structure
 - Every operator in the tree can be represented as a timed automaton with input and output
 - The product of them results into a timed automaton again

- Specs:
Visit A before B and visit B within $[l, r]$
- MTL:
 $\phi = (\neg BUA) \wedge (\Diamond_{[l,r]} B)$
- Tree:



Resulting Path



- Generated timed automata and the fastest path using UPPAAL¹³

13. G. Behrmann, A. David, K. G. Larsen, J. Hakansson, P. Petterson, W. Yi, and M. Hendriks, "UPPAAL 4.0," in *Third International Conference on Quantitative Evaluation of Systems, 2006. QEST 2006.*, 2006, pp. 125–126.

Robot Dynamics and Control

We consider a non-holonomic unicycle robot dynamics as given below:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = u \begin{bmatrix} \cos \theta \\ \sin \theta \\ 0 \end{bmatrix} + w \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

u and w are the control inputs.

Since we are dealing with time-bounded motion planning, we represent state and time pair by (q, t) where $q = [x, y, \theta]$.

- **Lemma:** If $\bar{u}(t)$ and $\bar{w}(t)$ drives the dynamics from the state $(q_0, 0)$ to $(q_1, 1)$, then,

$$u(t_0, t) = \frac{1}{\lambda} \bar{u}(t) \text{ and } w(t_0 + t) = \frac{1}{\lambda} \bar{w}(t)$$

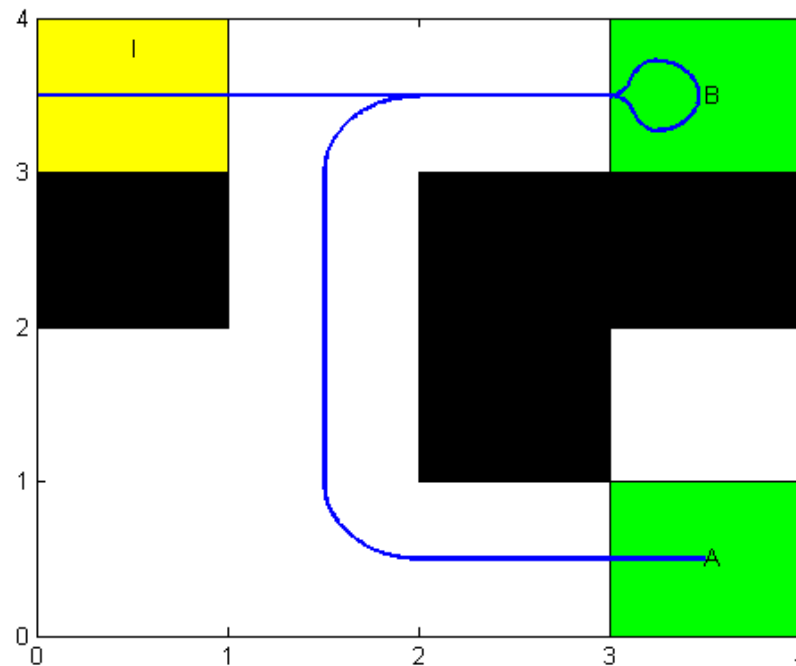
moves the system from (q_0, t_0) to $(q_1, t_0 + \lambda)$ for any $\lambda > 0$.

Moreover, if \bar{u} and \bar{w} move the system optimally, i.e.

$$J(\bar{u}, \bar{w}) = \min_{\{u(\cdot), w(\cdot)\}} \int_0^1 [ru^2(t) + w^2(t)] dt$$

Then, u and w given above are also optimal for moving the system from (q_0, t_0) to $(q_1, t_0 + \lambda)$.

- Task: $\phi = (\neg A \cup B) \wedge (\Diamond B)$



The robot starts from the initial position I (yellow block) and according to the task, it visits B before visiting A .

Complexity Analysis

Table: Computation Time for Typical MITL Formula

MITL Formula	Map Grid size	Transformation Time (in s)	Number of Transitions	Synthesis Time (in s)
ϕ_1	2x2	< 0.001	22	0.016
ϕ_2	2x2	0.004	69	0.018
ϕ_3	2x2	0.40	532	0.10
ϕ_4	2x2	0.46	681	0.12
ϕ_1	4x4	0.004	181	0.062
ϕ_1	8x8	0.015	886	0.21
ϕ_2	8x8	0.015	1795	0.32

$$\phi_1 = (\neg A \cup B) \wedge (\Diamond A)$$

$$\phi_2 = \Box \Diamond_{[0,2]} A$$

$$\phi_3 = \Diamond_{[0,4]} A \wedge \Diamond_{[0,4]} B$$

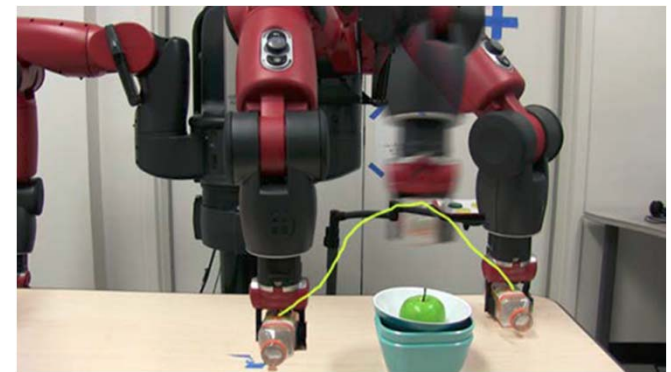
$$\phi_4 = \Diamond_{[2,4]} A \wedge \Diamond_{[0,2]} B$$

■ Summary

- Expressed motion planning problem with metric temporal logic for task planning
- Proposed two methods to solve motion planning problem with timed temporal logic constraints
 - Convert temporal constraints to mixed integer constraints
 - Convert temporal constraints to timed automata
 - Implemented a tool chain to convert timed temporal specifications to a timed automaton
 - The method is scalable as shown in the previous slide

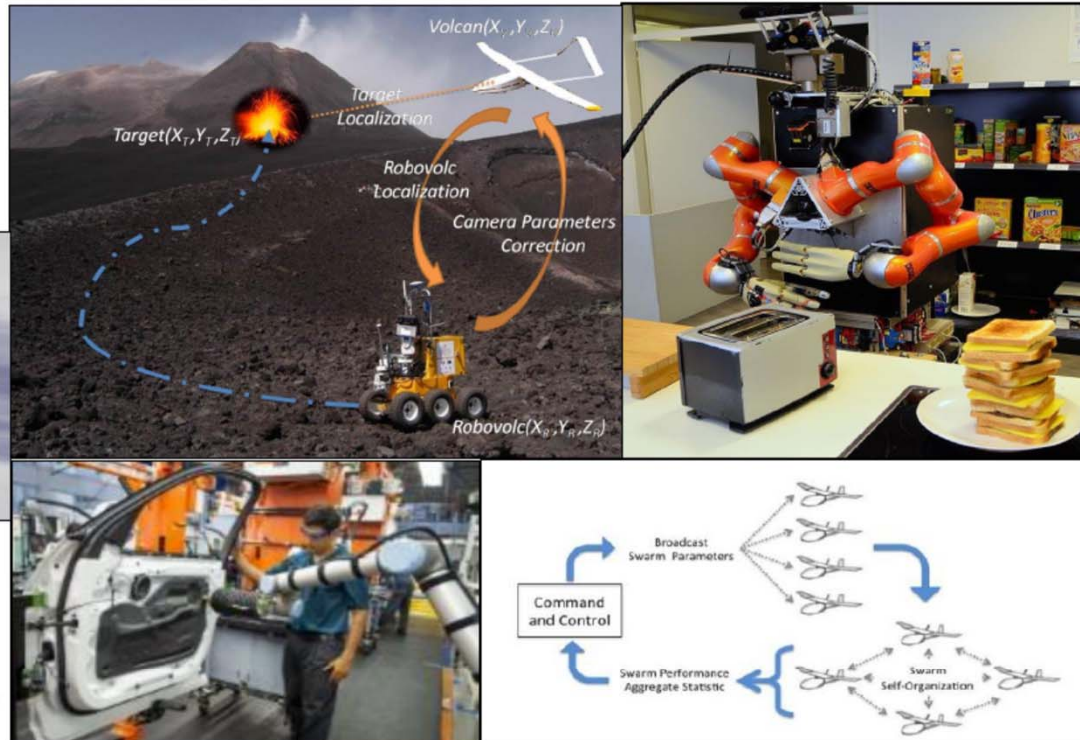
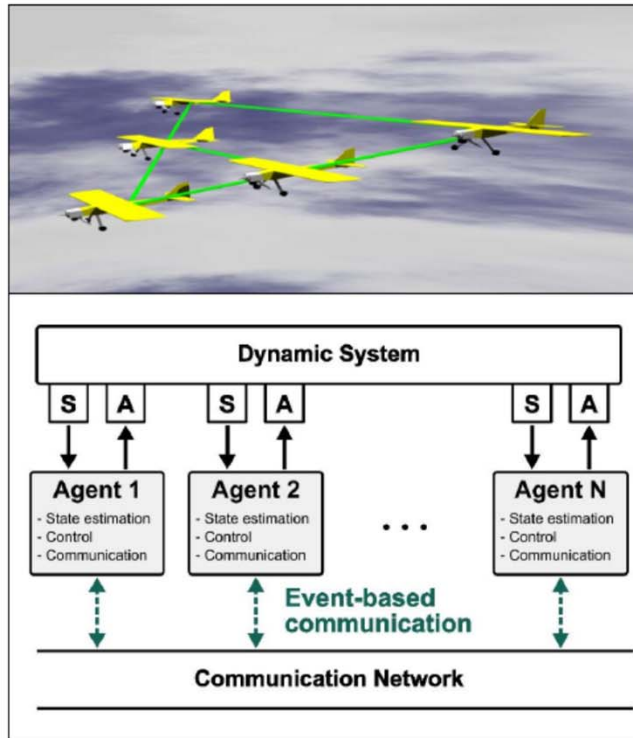
■ Future work

- Extend to robotic manipulation tasks
- Bridge the gap between action grammars and motion planning / controls
- Extend to multi-agent tasks
- Complexity reduction



Part III. Event –Triggered Controller Synthesis for Dynamical Systems with Temporal Logic Constraints

Motivation



The finite time logical constraints may arise due to the complex task description or decision making process, while the information constraints emerge as a consequence of the limitations on communication and computation capabilities.

Optimization Problem:

$$\begin{aligned} & \min_u J(x, u) \\ & \text{subject to } \dot{x} = f(t, x, u) \\ & \quad L_i(x[t_0]) = \text{true} \quad (x[t_0] \models \varphi) \quad i = 1, 2, \dots, l \\ & \quad g_j(t, x, u) \leq 0 \quad j = 1, 2, \dots, k \end{aligned}$$

Or feasibility problem

$$\begin{aligned} & \text{find } u \\ & \text{subject to } \dot{x} = f(t, x, u) \\ & \quad L_i(x[t_0]) = \text{true} \quad (x[t_0] \models \varphi) \quad i = 1, 2, \dots, l \\ & \quad g_j(t, x, u) \leq 0 \quad j = 1, 2, \dots, k \end{aligned}$$

u is an event-triggered controller, i.e., it must not require continuous feedback of the measurements.

Let $\Pi = \{\pi_1, \pi_2, \dots, \pi_p\}$: Set of Atomic Propositions, with each π_i , \exists a labelling function $\mathcal{L}_i : \mathcal{X} \rightarrow \{0, 1\}$. $\pi_i \subseteq \mathcal{X}$ For finite time logic, $\pi_i \subseteq \tilde{\mathcal{X}} = \mathcal{X} \times [0, T]$

Definition

LTL Syntax: The syntax of LTL formulas are defined according to the following grammar rules:

$$\phi ::= \top \mid \pi \mid \neg\phi \mid \phi \vee \phi \mid \phi \mathbf{U} \phi \mid \phi \mathbf{R} \phi$$

Definition

LTL Semantics The semantics of any formula ϕ over the trajectory $x[t_0]$ is recursively defined as:

$$x[t_0] \models \pi \text{ iff } \mathcal{L}_\pi(x(t_0)) = 1$$

$$x[t_0] \models \phi_1 \wedge \phi_2 \text{ iff } x[t_0] \models \phi_1 \text{ and } x[t_0] \models \phi_2$$

$$x[t_0] \models \phi_1 \mathbf{U} \phi_2 \text{ iff } \exists s \geq t_0 \text{ s.t. } x[s] \models \phi_2 \text{ and } \forall t_0 \leq s' < s, x[s'] \models \phi_1.$$

$$x[t_0] \models \phi_1 \mathbf{R} \phi_2 \text{ iff } \forall s \geq t_0 x[s] \models \phi_2 \text{ or } \exists s' \text{ s.t. } t_0 \leq s' < s, x[s'] \models \phi_1.$$

Controllers for Temporal Logic Constraints

- Many existing works have proposed methods to design a controller to satisfy temporal logic constraints.
- To mention a few amongst them:
 - Belta et al (IEEE TRO 2007), Kress-Gazit et al (IEEE TRO 2009), Fainekos et al (Automatica 2009), Doherty et al (Autonomous Agents and Multi-Agent Systems 2009), Guo et al (ICRA 2013), Lindemann et al (ACC 2017)... and the list continues.
- However, the controller designed in this framework are *feedback* in nature.

New Approach

Two Sub-Problems (A Compositional design approach)

- ① Design a feedback controller ($\gamma(x(t))$) controller to satisfy $L_i(x[t_0])$.
- ② Replace $\gamma(x(t))$ by a suitable event-triggered controller.
 - approximate the feedback trajectory as closely as possible

Event Based ϵ -close Controller Synthesis

$$\dot{x} = f_0(t, x) + \sum_{i=1}^m f_i(t, x) \cdot u_i$$
$$x(t_0) = x_0.$$

where $x(t) \in \mathcal{X} \subseteq \mathbb{R}^n$, $u_i(t) \in \mathbb{R}$. Let $u_i(t) = \gamma_i(x(t))$ generates the trajectory x_c .

For a given $\epsilon > 0$, the objective is to replace the feedback controller $\gamma_i(x(t))$ with an event based controller of the form $\gamma_i^e(x(t_k))$ such that

$$\sup_t \|x_e(t) - x_c(t)\| \leq \epsilon$$

Assumptions

(A1) $\gamma_i(\cdot)$ and for all t , $f_i(t, \cdot)$ are Lipschitz functions with Lipschitz constants L_γ^i , L_f^i respectively, for all $i = 0, 1, 2, \dots, m$.

(A2) $(f_i(t, x)\gamma_i(x))_x(t, x) = \frac{\partial(f_i(t, x)\gamma_i(x))}{\partial x}$ are Lipschitz continuous with Lipschitz constants L_1^i and $\frac{\partial(f_0(t, x))}{\partial x}$ is Lipschitz continuous with constant L_1^0 .

(A3) $y(t)$ is exponentially stable with

$$\dot{y} = A(t)y \quad (3)$$

where $A(t) = \frac{\partial F(t, x)}{\partial x} \Big|_{x=x_c(t)}$.

$$F(t, x) = f_0(t, x) + \sum_{i=1}^m f_i(t, x)\gamma_i(x)$$

We design: $\forall t > t_k, \gamma_i^e(t) = \gamma_i(x(t_k))$ where t_k is the latest triggering instance. Let $e(t) = x_e(t) - x_c(t)$
 $\delta(t) = \sum_{i=0}^m f_i(t, x_e)(\gamma_i^e(x_e(t_k)) - \gamma_i(x_e(t)))$

Proposition

For all t ,

$$\|e(t)\|_2 \leq \frac{c_4}{2c_1} \int_{t_0}^t e^{-(t-s)c_3/2c_2} \|\delta(s)\|_2 ds \quad (4)$$

c_i depends on the characteristics of (3). (The proof is based on converse Lyapunov Theorem).

$$\|e(t)\|_2 \leq \frac{c_2 c_4}{c_1 c_3} \sup_{s \in [t_0, t]} \|\delta(s)\|_2$$

Event Generating Function

$$f_{event}(\|\delta\|_2) = \frac{c_1 c_3}{c_2 c_4} \epsilon - \|\delta\|_2$$

Event is generated when $f_{event} < 0$. This implies $\|e(t)\|_2 \leq \epsilon$ for all t .

$\therefore f_{event}(\cdot)$ and $\gamma_i^e(x(t_k))$ together ensure $\sup_t \|x_e(t) - x_c(t)\|_2 \leq \epsilon$.

Proposition (sufficiency)

$\forall t > t_k, \|x_e(t) - x_e(t_k)\| \leq \frac{\epsilon}{M}$ ensures $\|e(t)\| \leq \epsilon$, for some
 $M = \frac{c_2 c_4}{c_1 c_3} \sum_{i=1}^m \bar{M}_i L_{\gamma}^i$ and $\bar{M}_i = \sup_{x \in \Omega_{\epsilon}} f_i(t, x)$.

Proposition (No Zeno behavior)

Inter event time for the event triggering mechanism f_{event} is bounded from below by $T_{min} = \min_k(t_{k+1} - t_k)$:

$$e^{\bar{L}T_{min}} \int_0^{T_{min}} K(s + t_k) ds = \frac{\epsilon}{G_{\infty} L_{\gamma}}$$

$$\bar{L} = L_f + \sum_{i=0}^m L_g^i \|\gamma_i(x(t_k))\|_2,$$

$$K(s) = \left\| f_0(s, x(t_k)) + \sum_{i=0}^m \gamma_i(x(t_k)) f_i(s, x(t_k)) \right\|_2,$$

$$\sum_{i=0}^m \|f_i(t, x)\|_2 \leq G_{\infty} < \infty \text{ and, } L_{\gamma} = \max\{L_{\gamma}^i \mid i = 1, 2, \dots, m\}.$$

Event-Triggered Controller and Temporal Logic

So far given certain continuous feedback control laws,
an event triggered control law can be designed to
approximate the actual trajectory arbitrary closely.

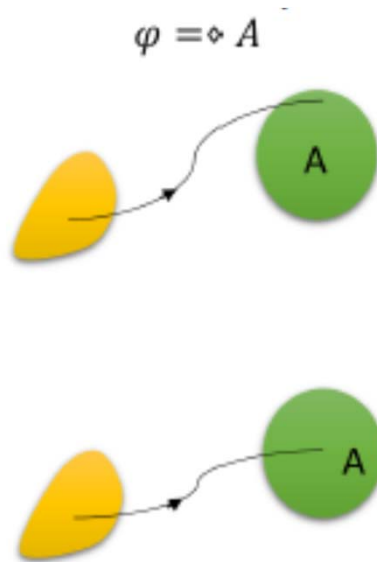
Is it sufficient? **No.**

Event-Triggered Controller and Temporal Logic

So far given certain continuous feedback control laws,
an event triggered control law can be designed to
approximate the actual trajectory arbitrary closely.

Is it sufficient? **No.**

Here is an example:

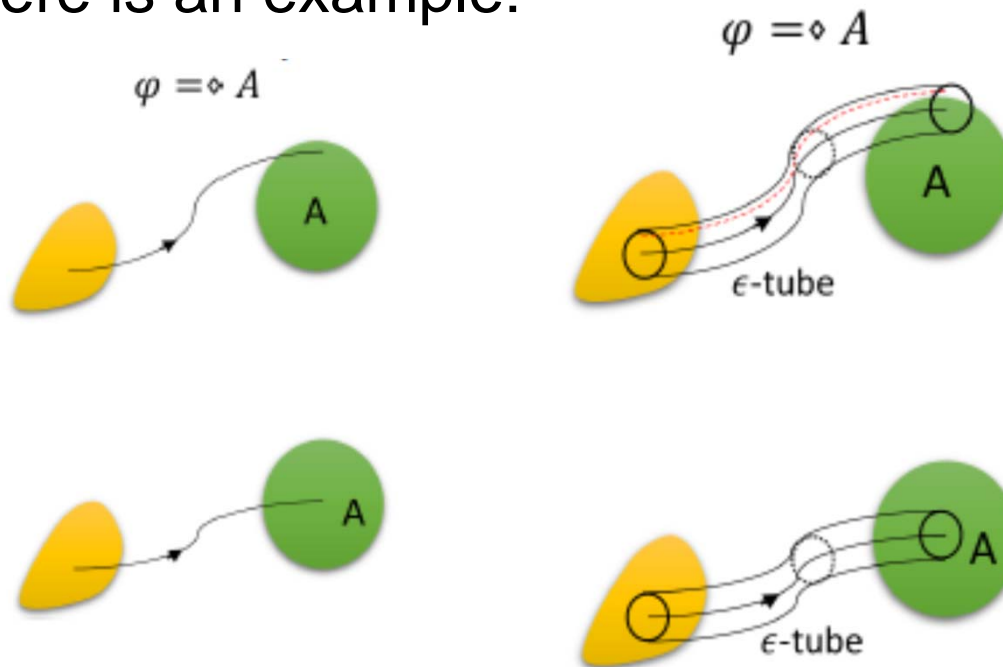


Event-Triggered Controller and Temporal Logic

So far given certain continuous feedback control laws,
an event triggered control law can be designed to
approximate the actual trajectory arbitrary closely.

Is it sufficient? **No.**

Here is an example:

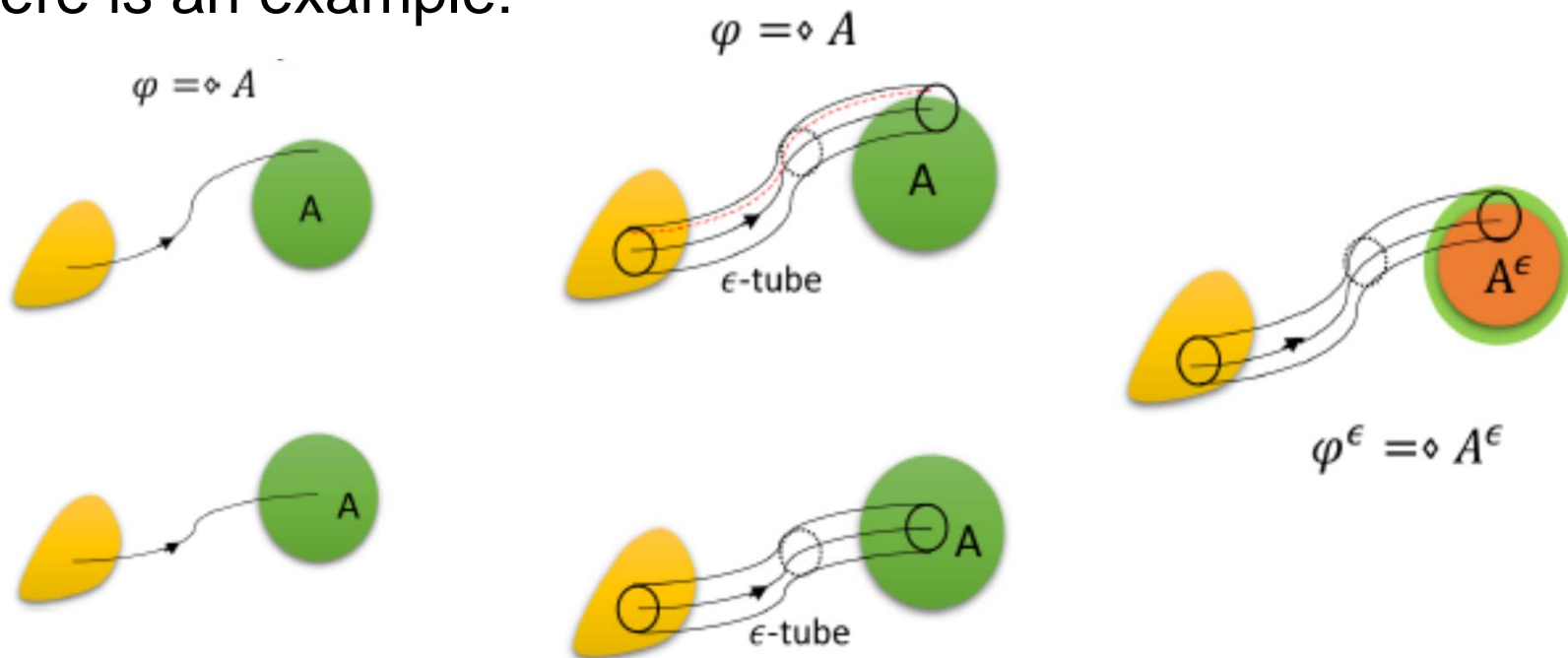


Event-Triggered Controller and Temporal Logic

So far given certain continuous feedback control laws,
an event triggered control law can be designed to
approximate the actual trajectory arbitrary closely.

Is it sufficient? **No.**

Here is an example:



Robust Temporal Logic Formula (Fainekos 2008)

Definition

In a given metric space (\mathcal{X}, ρ) the open ball centered at $x \in \mathcal{X}$ of radius r is defined as $B_x(r) = \{y \in \mathcal{X} \mid \rho(x, y) < r\}$. Let $\epsilon > 0$ be a given parameter, then the ϵ -contraction of the set $\mathcal{Y} \subseteq \mathcal{X}$ is denoted by $\mathcal{Y}^\epsilon = \{y \in \mathcal{Y} \mid B_y(\epsilon) \subseteq \mathcal{Y}\}$.

Similarly, the ϵ -expansion of the set is denoted by $\mathcal{Y}^{-\epsilon} = \{x \in \mathcal{X} \mid \exists y \in \mathcal{Y}, x \in B_y(\epsilon)\}$.

For a formula φ , the ϵ -robust formula is constructed as follows:

- 1) replace each $\pi_i(\subseteq \mathcal{X})$, which is not preceded by any negation, by π_i^ϵ .
- 2) any π_j that is preceded by a negation (\neg) should be replaced by $\pi_j^{-\epsilon} \cup (\mathcal{X} \setminus \mathcal{X}^\epsilon)$. When $\mathcal{X} = \mathcal{X}^\epsilon$ for all finite $\epsilon > 0$ (e.g. $\mathcal{X} = \mathbb{R}^n$), $\pi_j^{-\epsilon} \cup (\mathcal{X} \setminus \mathcal{X}^\epsilon) = \pi_j^{-\epsilon}$.

Proposition

Let $x(\cdot) : [t_0, T) \rightarrow \mathcal{X}$ be a curve and the corresponding trajectory $x[t_0]$ satisfies the TL formula φ^ϵ for some $\epsilon > 0$. Then for all $\delta \leq \epsilon$ and for any curve $y(\cdot) : [t_0, T) \rightarrow \mathcal{X}$ such that $\sup_t \rho(x(t), y(t)) \leq \delta$, $y[t_0] \models \varphi$.

Remark

In order to construct a π_i^ϵ , π_i must have a non-empty interior (its interior must contain a ball of radius ϵ).

Framework for the Controller Synthesis

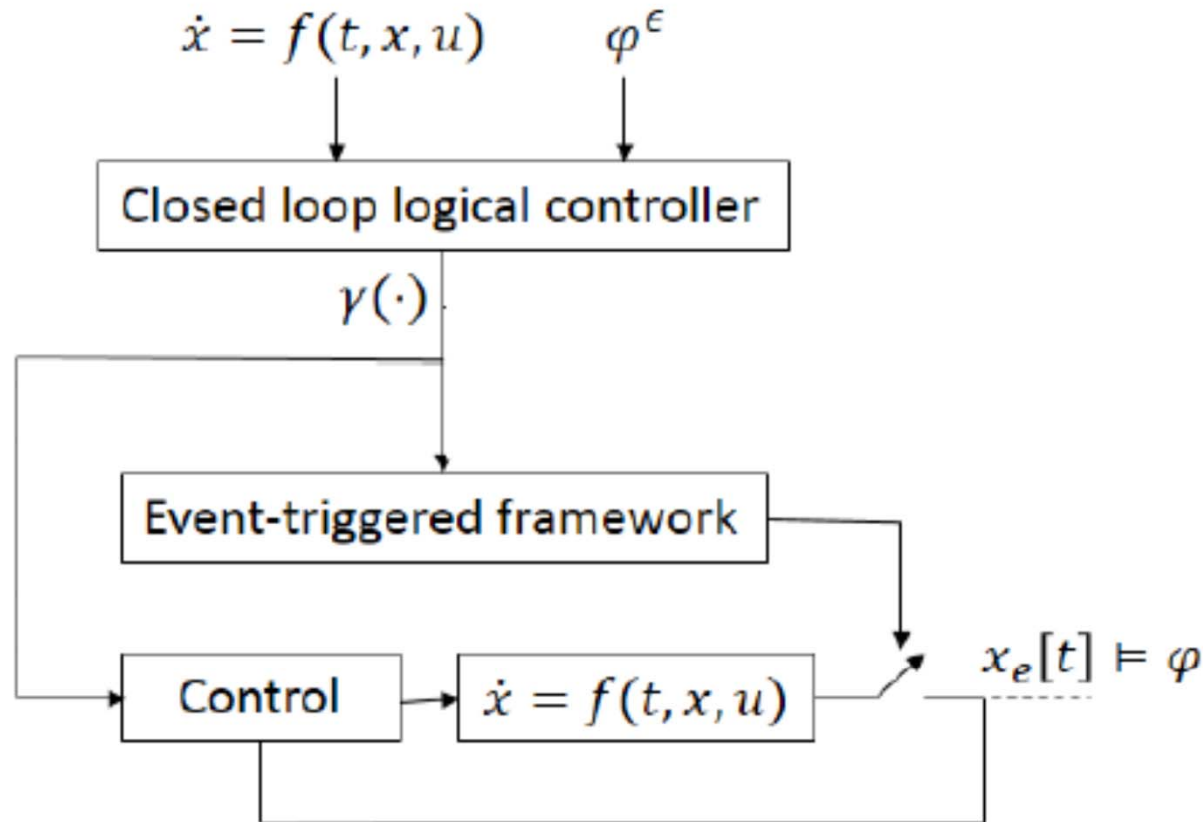
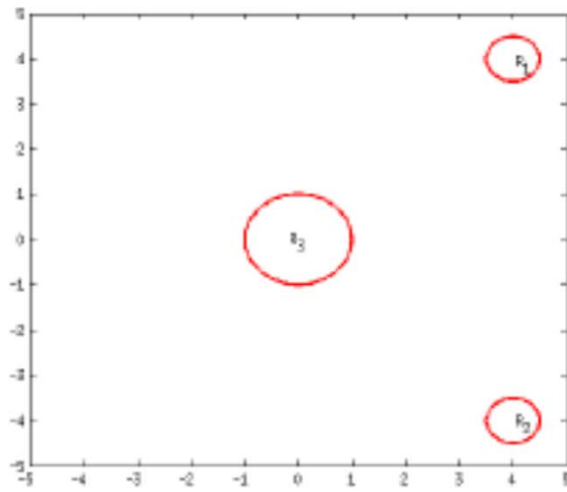


Figure : Schematic of two-step event-triggered controller synthesis with logical constraints

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos \theta \\ \sin \theta \\ 0 \end{bmatrix} v + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} w \quad (7)$$

where $x, y \in \mathbb{R}^2$ is the physical position of the robot (w.r.t. certain global frame) and $\theta \in [0, 360^\circ)$ is the heading angle.



The task is given as follows:

$$\varphi \models \Diamond \pi_2 \wedge (\neg \pi_2 \mathbf{U} \pi_1) \wedge \Box \neg \pi_3 \quad (8)$$

The formula defines the task of avoiding R_2 until reaching R_1 and eventually reaching R_2 , and during the whole time the trajectory should avoid R_3 .

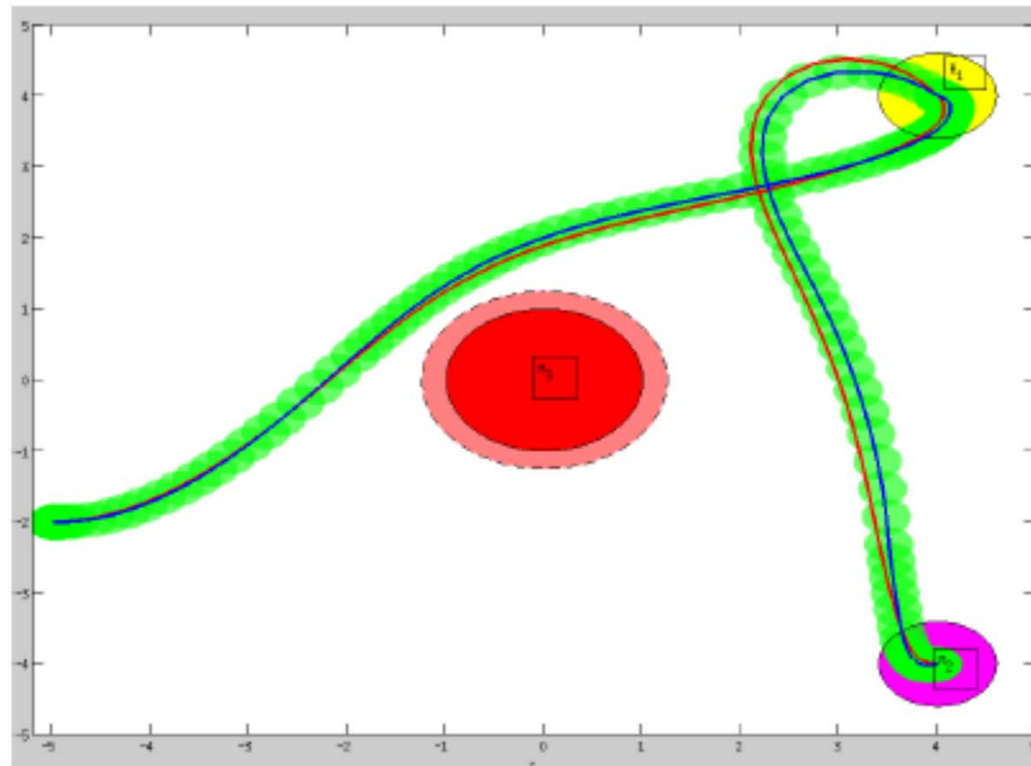


Figure : The closed loop trajectory is plotted using blue line and the event triggered trajectory with red line. The green tube around the nominal trajectory has radius 0.25. The initial position and orientation of the robot is $(-5, -2, 0)$

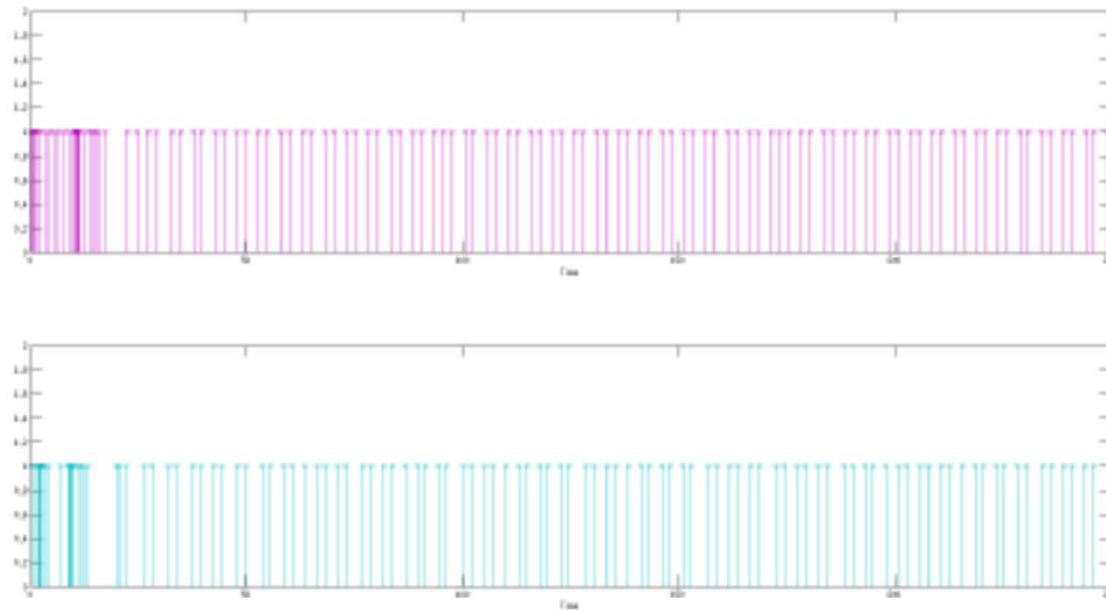


Figure : The upper graph shows the triggering instances for the trajectory from initial position to R_1 . The lower graph shows the same for the other segment of the trajectory.

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = - \begin{bmatrix} \sin(x_1) \\ x_2 \end{bmatrix} + \begin{bmatrix} -x_2 \\ x_1 \end{bmatrix} u$$

where $x_1(0) = 0, x_2(0) = 1$. $\pi_1 = B_0(0.1) \subset \mathbb{R}^2$. $\varphi = \Diamond \Box \pi_1$, $\gamma(x(t)) = -x_2(t)$, and we consider π_1^ϵ with $\epsilon = 0.05$.

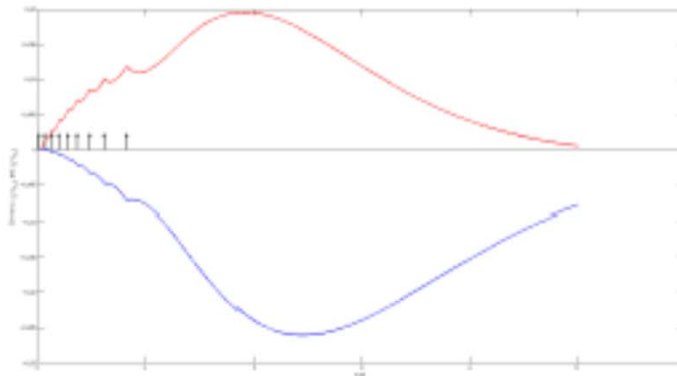


Figure : Error components and the triggering instances.

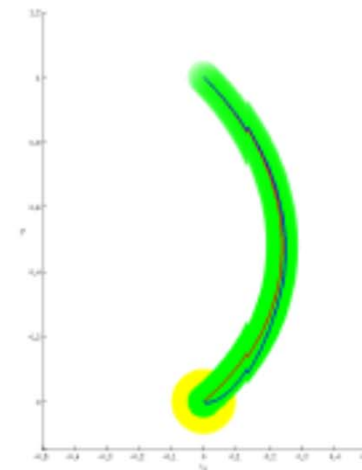


Figure : Actual trajectory and event-based trajectory.

Summary

Framework for integrating event-triggered controller synthesis and logic based controller synthesis.

Explicit event triggering condition derived, and that condition is sufficient for ensuring the event-triggered trajectory will not deviate more than ϵ amount from the ideal closed loop continuous trajectory.

Simulation results show significant reduction in communicating the state value for updating the controller.

This reduces the communication and computation costs.

Part IV. Event –Triggered Feedback Control for Signal Temporal Logic Tasks

- Introduction
- Signal Temporal Logic
- Event based Controller for Temporal Logic Tasks
 - Problem Formulation
 - Proposed Problem Solution
- Summary

Systems under Complex Tasks



Photo Credit: Håkan Lindgren



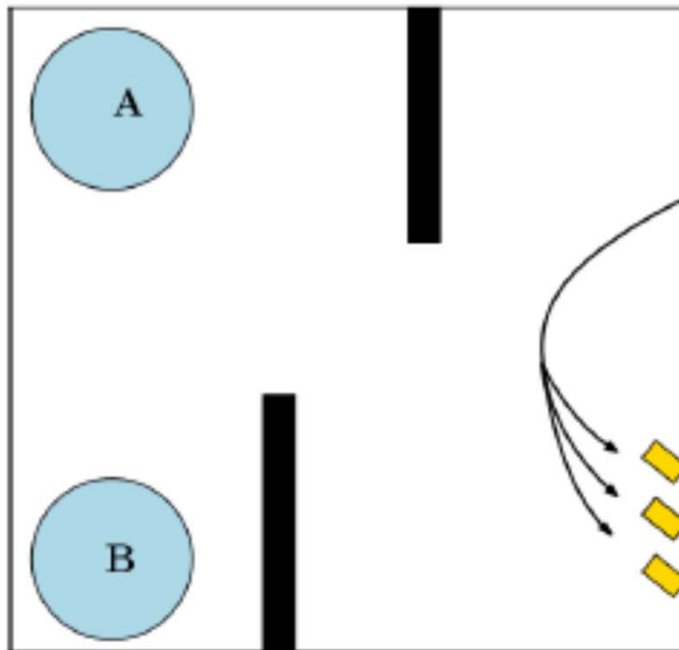
Photo Credit: <https://www.volvotrucks.com>

- **Spatial tasks:** reachability or sequencing
- **Collaborative spatial tasks:** connectivity maintenance, consensus, formation control, surveillance, coverage
- **Timed tasks,** i.e., spatial tasks under given deadlines
- **Safety tasks:** collision or region avoidance

Temporal Logics

- **Temporal Logics** allow to mathematically express complex tasks that are imposed on multi-agent systems.

Environment scattered with obstacles



Group of Nexus 4WD Robots: $\dot{x} = f(x, u)$



Complex Task:

- ♦ **R1** goes to **A** in no more than 2 minutes
- ♦ **R2** keeps a minimum distance to **R1** and **R3**
- ♦ **R3** goes to **B** in no more than 2 minutes
- R1, R2, and R3** avoid collisions

Existing Frameworks for Controllers under Temporal Logic Tasks

- Automata/abstraction-based approaches ^{1 2}
- Game-based approaches ^{3 4}
- Optimization-based approaches ^{5 6}
- Learning-based approaches ⁷

¹ M. Kloetzer and C. Belta. *IEEE Transactions on Robotics*. 2010.

² S.G. Loizou and K.J. Kyriakopoulos. *IEEE Conference on Decision and Control (CDC)*. 2004.

³ H. Kress-Gazit, G.E. Fainekos, and G.J. Pappas. *IEEE Transactions on Robotics*. 2009.

⁴ S. Dathathri, R.M. Murray. *IEEE Conference on Decision and Control (CDC)*. 2017.

⁵ Z. Liu, B. Wu, J. Dai, and H. Lin. *IEEE Conference on Decision and Control (CDC)*. 2017.

⁶ S. Karaman and E. Frazzoli. *International Journal of Robust and Nonlinear Control*. 2011.

⁷ P. Schillinger, M. Bürger, and D.V. Dimarogonas. *IEEE International Conference on Robotics and Automation (ICRA)*. 2018.

New Approach

- Existing controllers are feedback in nature { generally requires expensive communication & sensing resources.
- Abstraction based approaches generally suffer from state explosion -- computationally expensive.
- Inability to incorporate time constraints in many existing approaches.

We use **Signal Temporal Logic** and derive event-triggered control strategies.

Key features: state- and time-constrained tasks, robust, computationally-efficient (abstraction-free), inexpensive implementation (event-trigger communication & control)

- Predicates⁸ μ are obtained after evaluation of a predicate function $h : \mathbb{R}^n \rightarrow \mathbb{R}$ as

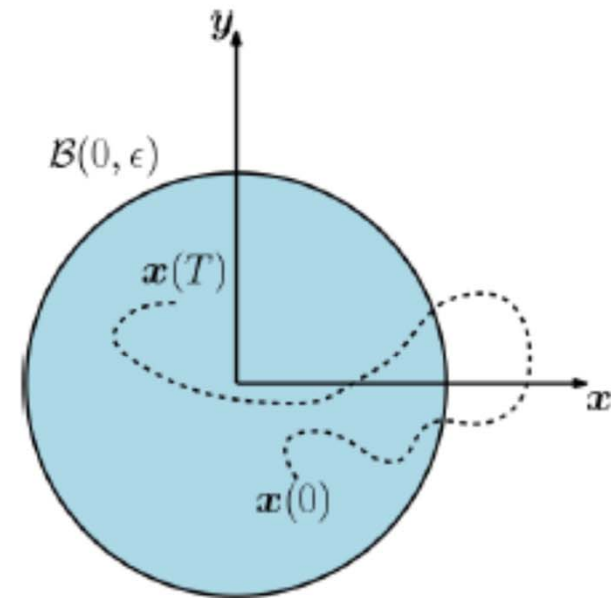
$$\mu := \begin{cases} \top & \text{if } h(\mathbf{x}) \geq 0 \\ \perp & \text{if } h(\mathbf{x}) < 0 \end{cases}$$

Example:

- Assume the predicate $\|\mathbf{x}(\tau)\| \leq \epsilon$.
- The predicate function

$$h(\mathbf{x}(\tau)) := \epsilon - \|\mathbf{x}(\tau)\|$$

indicates $\|\mathbf{x}(\tau)\| \leq \epsilon$ iff $h(\mathbf{x}(\tau)) \geq 0$.



⁸O. Maler and D. Nickovic. *Int. Conference on Formal Modeling and Analysis of Timed Systems*, 2004.

- Syntax of STL⁸:

$$\phi ::= \top \mid \mu \mid \neg\phi \mid \phi(1) \wedge \phi(2) \mid \phi(1) U_{[a,b]} \phi(2) \mid F_{[a,b]}\phi \mid G_{[a,b]}\phi$$

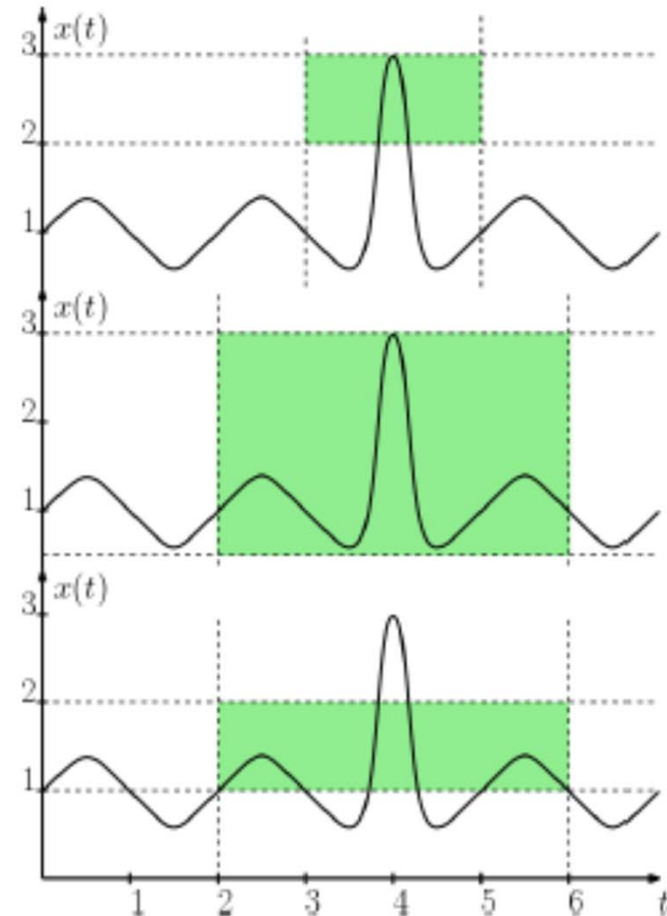
- **Semantics:** $(x, 0) \models \phi$ indicates that the signal $x : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}^n$ satisfies ϕ
- **Robust Semantics:** $\rho^\phi(x, 0)$ indicates how robustly the signal $x : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}^n$ satisfies ϕ

It holds that $(x, 0) \models \phi$ if $\rho^\phi(x, 0) > 0$.⁹

⁸ O. Maler and D. Nickovic. *Int. Conference on Formal Modeling and Analysis of Timed Systems*. 2004.

⁹ G.E. Fainekos and G.J. Pappas. *Theoretical Computer Science*. 2009.

- $\phi_1 = F_{[3,5]}(2 \leq x \leq 3)$ holds
and $\boxed{\rho^{\phi_1}(x, 0) = 0.5}$
- $\phi_2 = G_{[2,6]}(0 \leq x \leq 6)$ holds
and $\boxed{\rho^{\phi_2}(x, 0) = 0.01}$
- $\phi_3 = G_{[2,6]}(1 \leq x \leq 2)$ does not
hold and $\boxed{\rho^{\phi_3}(x, 0) = -1}$



Design Event-Triggered Control

Problem

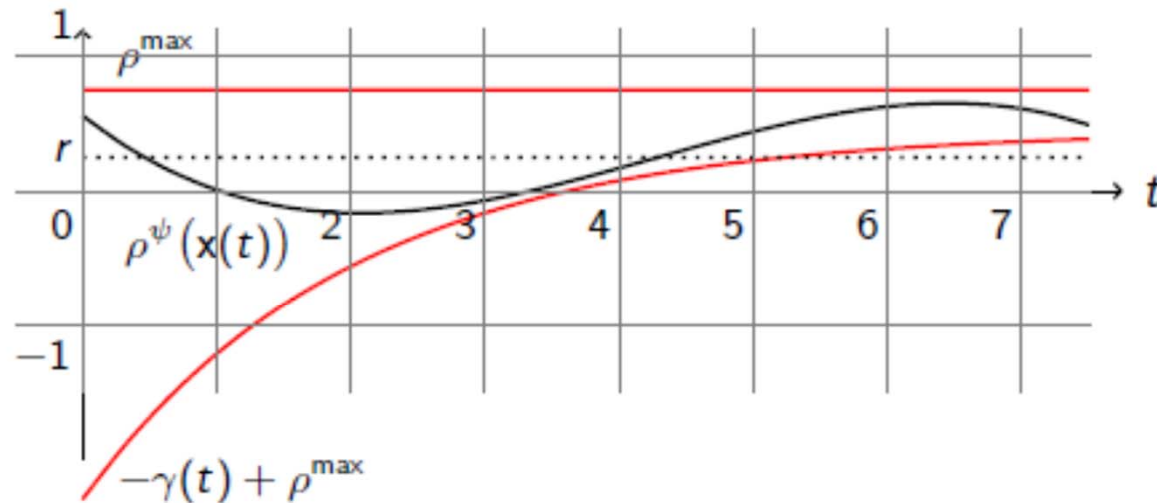
Given a dynamical system $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}, \mathbf{w})$ and an STL task ϕ , the objective is to design a **event-triggered** control law which ensures that the dynamical system satisfies the given STL task, i.e., $(\mathbf{x}, 0) \models \phi$

\mathbf{w} is a disturbance of bounded amplitude.

Approach

- First, we use a prescribed performance control (PPC)¹⁰ law

$$u(x, t) := -L(x) \left(f(x) - \frac{\frac{\partial \rho^\psi(x)}{\partial x}}{\left\| \frac{\partial \rho^\psi(x)}{\partial x} \right\|^2} (\xi \dot{\gamma} - k\epsilon) \right)$$



¹⁰ C. Bechlioulis and G. Rovithakis. *Automatica*, 2014.

- We replace the feedback control a event-triggered control¹⁰:

Theorem

The dynamical system satisfies the STL task ϕ under the PPC framework and if the event-triggered control law \hat{u} has the form

$$\hat{u}(t) := u(x(t_i), t) \quad \forall t \in (t_i, t_{i+1}] \quad (1)$$

where the triggering instances $\{t_i\}$ are generated as:

$$\begin{aligned} t_0 &:= 0 \\ t_{i+1} &:= \inf\{t > t_i \mid \|x(t) - x(t_i)\| > \delta_x\} \quad i \geq 1, \end{aligned} \quad (2)$$

for some $\delta_x > 0$.

¹⁰L. Lindemann, D. Maity, J. Baras, D. Dimarogonas. Conference on Decision and Control, 2018

Experimental Result

- Consider the agent dynamics¹¹

$$\dot{\mathbf{x}}^i = g_i(\mathbf{x}^i) \mathbf{u}^i = \begin{bmatrix} \cos(x_3^i) & -\sin(x_3^i) & 0 \\ \sin(x_3^i) & \cos(x_3^i) & 0 \\ 0 & 0 & 1 \end{bmatrix} (B_i^T)^{-1} R_i \mathbf{u}^i,$$

where R_i is the wheel radius and

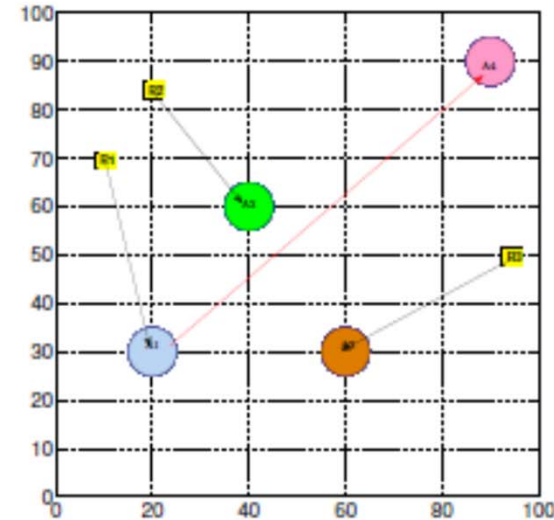
$$B_i := \begin{bmatrix} 0 & \cos(\pi/6) & -\cos(\pi/6) \\ -1 & \sin(\pi/6) & \sin(\pi/6) \\ L_i & L_i & L_i \end{bmatrix}$$

describes geometrical constraints with L_i as the radius of the robot body. We set $L_i := 0.2$ and $R_i := 0.02$ for $i \in \{1, 2, 3\}$.

¹¹ Y. Liu, J. J. Zhu, R. L. Williams, and J. Wu. Robotics and Autonomous Systems, 2008

Experimental Result

- R1: 1) Eventually Go to **A1** and $\theta_1 = 45^\circ$
2) go to **A4** and stay close to **R2**
- R2: 1) Eventually Go to **A2** and $\theta_1 = 45^\circ$
2) Stay close to **R1** and **R3**
- R3: 1) Eventually Go to **A3** and $\theta_1 = 45^\circ$
2) Stay close to **R3**



- Tasks:

$$\begin{aligned} \phi_1 &:= F_{[0,50]}((\|p_1 - A1\| \leq \epsilon) \wedge \|\theta_1 - 45^\circ\| \leq \epsilon)) \\ &\quad \wedge F_{[50,100]}((\|p_1 - A4\| \leq \epsilon) \wedge \|p_1 - p_2\| \leq \epsilon) \\ \phi_2 &:= F_{[0,50]}((\|p_2 - A2\| \leq \epsilon) \wedge (\|\theta_2 - 45^\circ\| \leq \epsilon)) \\ &\quad \wedge F_{[50,100]}(\|p_2 - p_3\| \leq \epsilon) \wedge \|p_2 - p_3\| \leq \epsilon \\ \phi_3 &:= F_{[0,50]}(\|p_3 - A3\| \leq \epsilon) \wedge (\|\theta_3 - 45^\circ\| \leq \epsilon)) \\ &\quad \wedge F_{[50,100]}(\|p_3 - p_2\| \leq \epsilon) \end{aligned}$$

Agent Trajectories

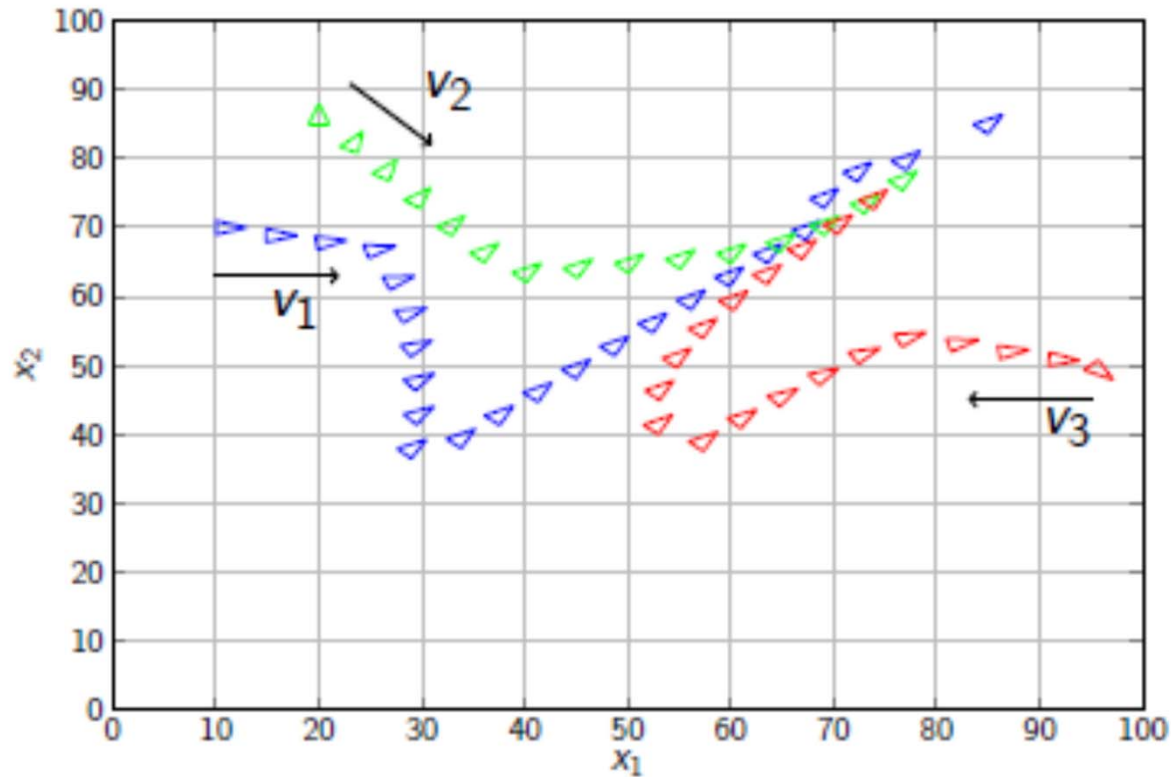


Figure : Agent Trajectories

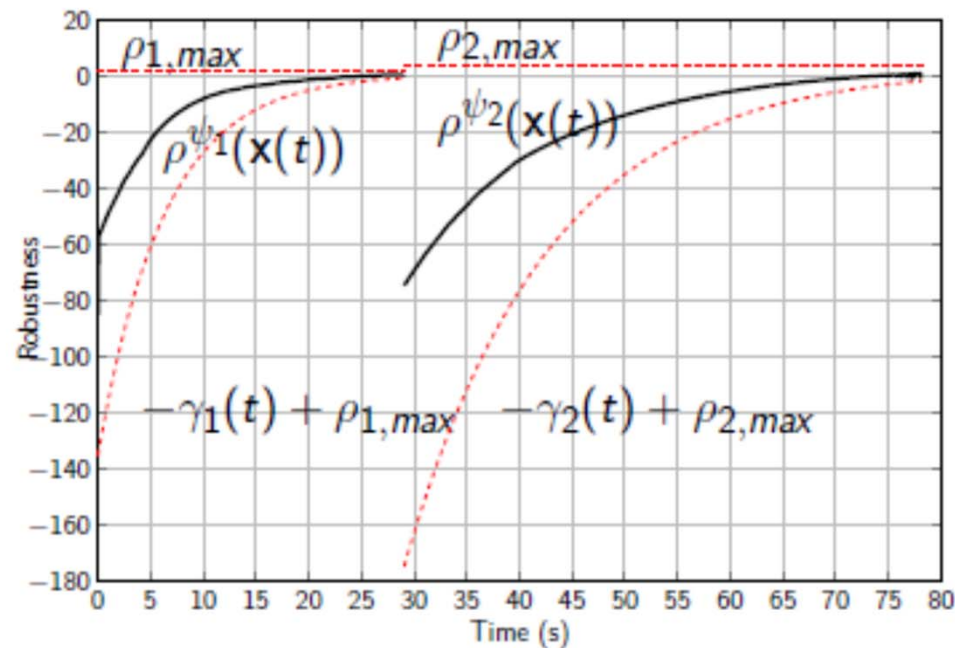


Figure : Robustness

- The experiment was implemented in 100 Hz frequency with a total of **7725** samples.
- Using our event-triggered feedback policy the control was updated only **185** times.

Summary:

- Abstraction-free, computationally-efficient, and robust method for bottom-up multi-agent systems
 - Robustness considered on two levels: robustness with respect to the task and with respect to disturbances
- Event-triggered control reduced the amount of communication significantly.

Summary:

- Abstraction-free, computationally-efficient, and robust method for bottom-up multi-agent systems
 - Robustness considered on two levels: robustness with respect to the task and with respect to disturbances
- Event-triggered control reduced the amount of communication significantly.

Future Work:

- Accounting for input limitations and dynamics
- Considering self-triggered control to reduce sensing.

1. Y. Zhou, J. Moschler, and J. S. Baras, “A System Engineering Approach to Collaborative Coordination of UAS’s in the NAS with Safety Guarantees”, *Proceedings of the 2013 Integrated Communications Navigation and Surveillance Conference (ICNS)*, pp. 1-12, Herndon, VA, April 8-10, 2014.
2. D. Maity and J. S. Baras "Motion Planning in Dynamic Environment with Bounded Time Temporal Logic Specifications", *Proceeding of the 23rd Mediterranean Conference on Control and Automotion (MED 2015)*, pp. 973-979, Torremolinos, Spain, June 16-19, 2015.
3. Y. Zhou and J. S. Baras, “Reachable Set Approach to Collision Avoidance for UAVs”, *Proceedings of 54th IEEE Conference on Decision and Control*, Osaka, Japan, December 15-18, 2015.
4. Y. Zhou, A. Raghavan and J. S. Baras, “Time Varying Control Set Design for UAV Collision Avoidance Using Reachable Tubes”, *Proceedings of 55th IEEE Conference on Decision and Control*, Las Vegas, USA, 2016.

5. Y. Zhou, D. Maity and J. S. Baras, “Optimal Mission Planner with Timed Temporal Logic Constraints”, *Proceedings of 2015 European Control Conference*, Linz, Austria, July 15-17, 2015.
6. Y. Zhou, D. Maity, and J. S Baras. “Timed Automata Approach for Motion Planning Using Metric Interval Temporal Logic.“, *Proceedings of 2016 European Control Conference*, Aalborg Denmark, June 29 - July 1, 2016.
7. D. Maity and J.S. Baras, “Event-Triggered Controller Synthesis for Dynamical Systems with Temporal Logic Constraints”, *Proceedings 2018 American Control Conference*, Milwaukee, USA, June 27–29, 2018.
8. L. Lindemann, D. Maity, J. Baras, and D. Dimarogonas, “ Event-Triggered Feedback Control for Signal Temporal Logic Tasks,” to appear *Proceedings 58th IEEE Conference on Decision and Control*, Dec. 2018

Thank you!

baras@isr.umd.edu

301-405-6606

<http://dev-baras.pantheonsite.io/>

Questions?